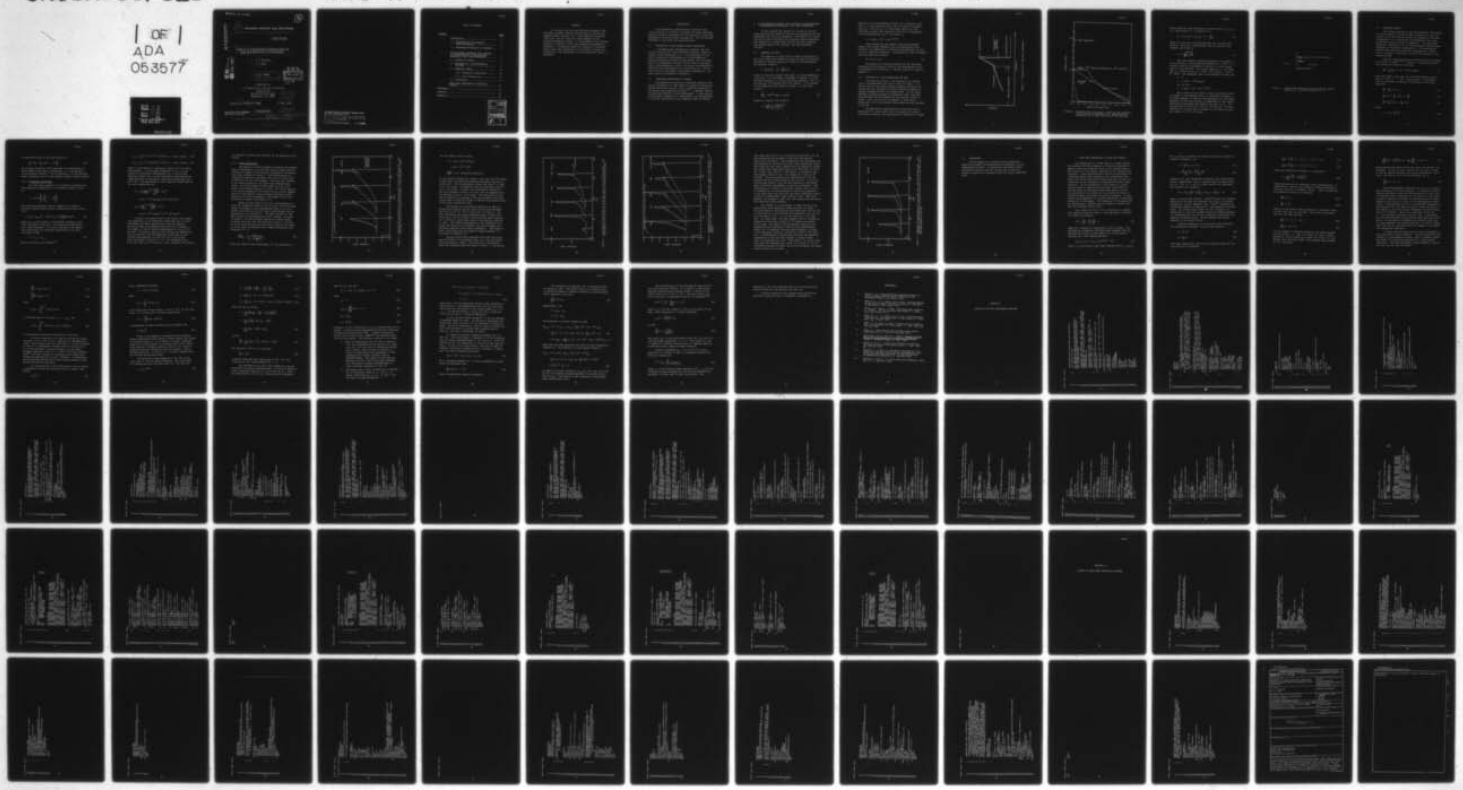
AD-A053 577 SYSTEMS SCIENCE AND SOFTWARE LA JOLLA CALIF F/G 21/2
ANALYSIS OF LASER-SUPPORTED DETONATION WAVES AND LASER BEAM PRO--ETC(U)
FEB 78 J R TRIPLETT, M H RICE F49620-77-C-0006
UNCLASSIFIED SSS-R-78-3564 AFOSR-TR-78-0775 NL

| OF |
ADA
053577



END
DATE
FILMED
6-78
DDC

(2)

SYSTEMS, SCIENCE AND SOFTWARE

(14) SSS-R-78-3564

(6) # ANALYSIS OF LASER-SUPPORTED DETONATION WAVES AND LASER BEAM PROPAGATION IN TWO DIMENSIONS.

(10) J. R. TRIPLETT
M. H. RICE

(9) FINAL REPORT.
1 Oct 76-30 Sep 77,

SUBMITTED TO:

AIR FORCE OFFICE OF SCIENTIFIC RESEARCH
BUILDING 410
BOLLING AIR FORCE BASE
WASHINGTON, D.C. 20332

(16) 2301
(17) 45

(15) CONTRACT No. F49620-77-C-0006

S^3 REF. 11056

(11) FEBRUARY 1978

(12) 82p.

(18) AFOSR (19) TR-78-0775

P. O. BOX 1620, LA JOLLA, CALIFORNIA 92038, TELEPHONE (714) 453-0060

388 507 JOB

AIR FORCE OFFICE OF SCIENTIFIC RESEARCH (AFSC)
NOTICE OF TRANSMITTAL TO DDC
This technical report has been reviewed and is approved for public release IAW AFR 190-12 (7b).
Distribution is unlimited.
A. D. BLOSE
Technical Information Officer

TABLE OF CONTENTS

ABSTRACT

It is shown that the Flux-Corrected Transport (FCT) algorithm correctly computes the propagation velocity and Chapman-Jouguet pressure for a laser-supported detonation (LSD) wave, even for zone size large compared to the radiation absorption length. A two-dimensional FCT computer program for analysis of LSD wave propagation phenomena is presented. A trajectory method for analysis of two-dimensional laser beam propagation in the paraxial Fresnel approximation, with continuously varying refractive index, is described.

1. INTRODUCTION

The developments described below constitute the initial portion of an effort to develop and apply efficient, versatile analytic techniques for the analysis of laser beam propagation and laser-target interactions. The two problems addressed in the report are described below.

1.1 EVALUATION OF NON-ADIABATIC SHOCK PROPAGATION

A two-dimensional hydrodynamics computer code was written using the FCT (Flux-Corrected Transport) method developed at NRL [References 1-5]. The performance of this code in describing the propagation of laser-supported detonation waves was studied in some detail from the standpoint of thermodynamic validity for the case of one-dimensional wave propagation; initial testing of two-dimensional detonation wave propagation was also carried out. The results in both cases appear to be highly satisfactory. The code listing is presented in Appendix I.

1.2 LASER-BEAM PROPAGATION IN PLASMAS

Two-dimensional methods for computing refraction and Fresnel diffraction of laser beams in plasmas and partially-ionized gases were devised, using a trajectory approach of the general type suggested by Glass.[6] A computer code employing these methods is under development. A code listing is presented in Appendix II, representing the implementation of the text formulation at an incomplete stage of testing.

2. FLUX-CORRECTED TRANSPORT (FCT) APPLIED TO ONE-DIMENSIONAL LASER-SUPPORTED DETONATION (LSD) WAVE PROPAGATION

In this section the results of a series of calculations using the FCT method to describe the propagation of one-dimensional LSD waves are presented. The object is to determine the accuracy with which the numerical method reproduces such details of a steady-state wave as the spike pressure, the Chapman-Jouguet (CJ) pressure, and the propagation velocity.

2.1 EQUATION OF STATE

The equation of state used for these calculations is that of an ideal gas modified to allow for single ionization. The specific internal energy is related to the temperature and specific volume by

$$E = \frac{1}{2} RT \left[f + 3x + \frac{2\varepsilon_o}{kT} x \right] , \tag{1}$$

where R is the gas constant (per gram), T is the temperature in kelvins, f is the number of degrees of freedom, k is the Boltzmann constant, ε_o is the ionization potential, and x is the fraction of molecules which are ionized. It is related to the temperature and specific volume, V, by the Saha equation

$$\frac{x^2}{1-x} = AVT^{3/2} \exp\left(-\varepsilon_o/kT\right) \tag{2}$$

where the constant A is given by

$$A = \frac{g_e g_i}{g_a} \frac{M}{N_o} \left[\frac{2\pi m_e k}{h^2} \right]^{3/2} ,$$

where M is the gram molecular weight, N_o is Avogadro's number, m_e is the electronic mass, and h is Planck's constant. With $g_e = 2$ and $g_i/g_a = 1.9$ (the weight factor for ions relative to atoms), the numerical value for A corresponding to a molecular weight of M = 28 g/mole is

$$A = 4.2662 \times 10^{-7} \text{ g cm}^{-3} \text{ K}^{-3/2}.$$

Given values for the specific volume and internal energy (the calling arguments from the hydrodynamic subroutine), the temperature is obtained from Equations (1) and (2) by an iterative procedure. The pressure is then calculated from the ideal gas relation,

$$PV = R(1 + x)T. \qquad (3)$$

The equation of state subroutine used for the above calculations also returns values of the sound speed for use in determining the time-step. A Fortran listing is given in Appendix I.

2.2 STRUCTURE OF A ONE-DIMENSIONAL LSD WAVE

A qualitative sketch of the structure of a steady-state, one-dimensional LSD wave is given in Figure 1. An initial shock at the spike pressure preheats and partially ionizes the gas, providing the free electrons necessary to initiate absorption from the incident beam. As the energy absorbed increases through the absorption front, the pressure decreases and the temperature and specific volume increase until the CJ point is reached, at which point essentially all of the incident beam energy has been absorbed by the gas.

The quantitative determination of the spike and CJ pressures for a given laser flux is indicated in Figure 2. The Hugoniot relation for conservation of energy for a steady

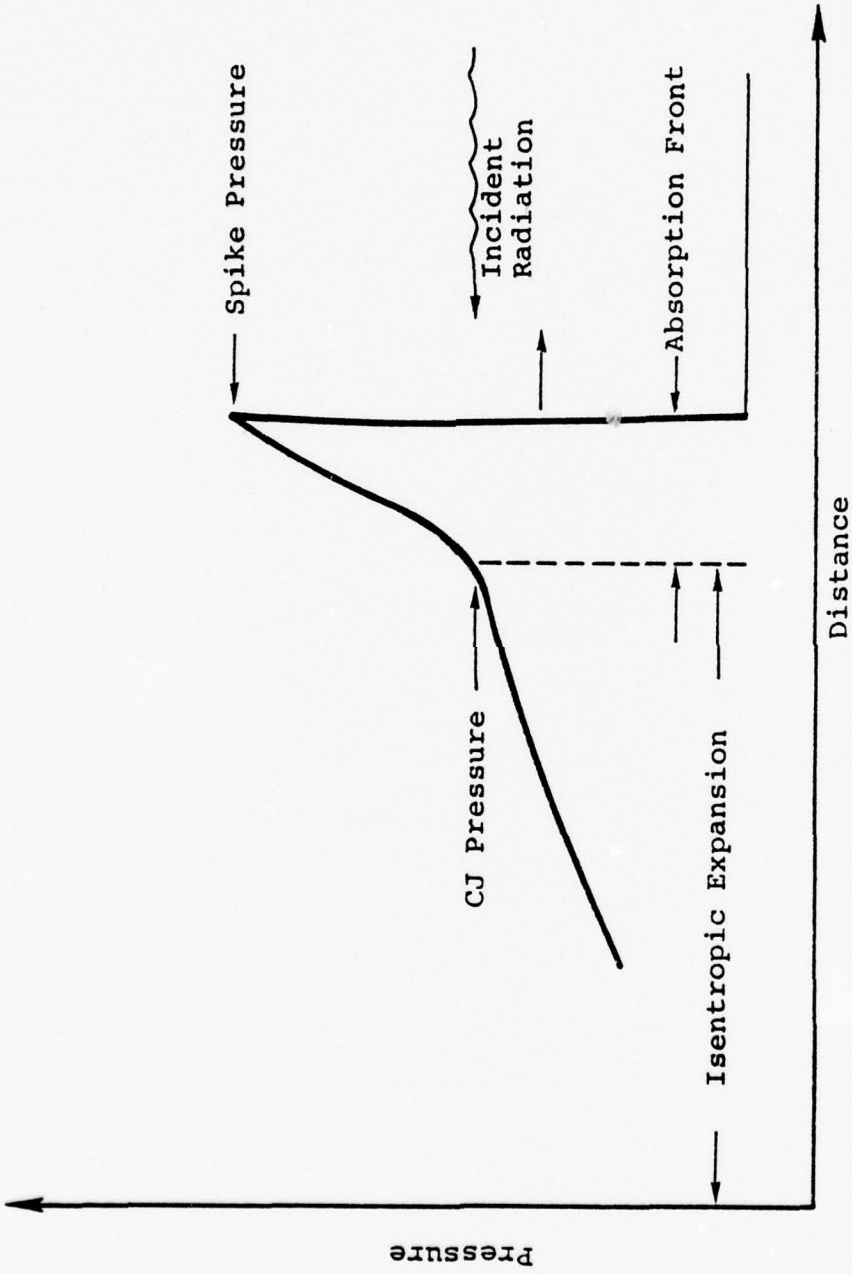


Figure 1. Structure of a LSD wave (schematic).

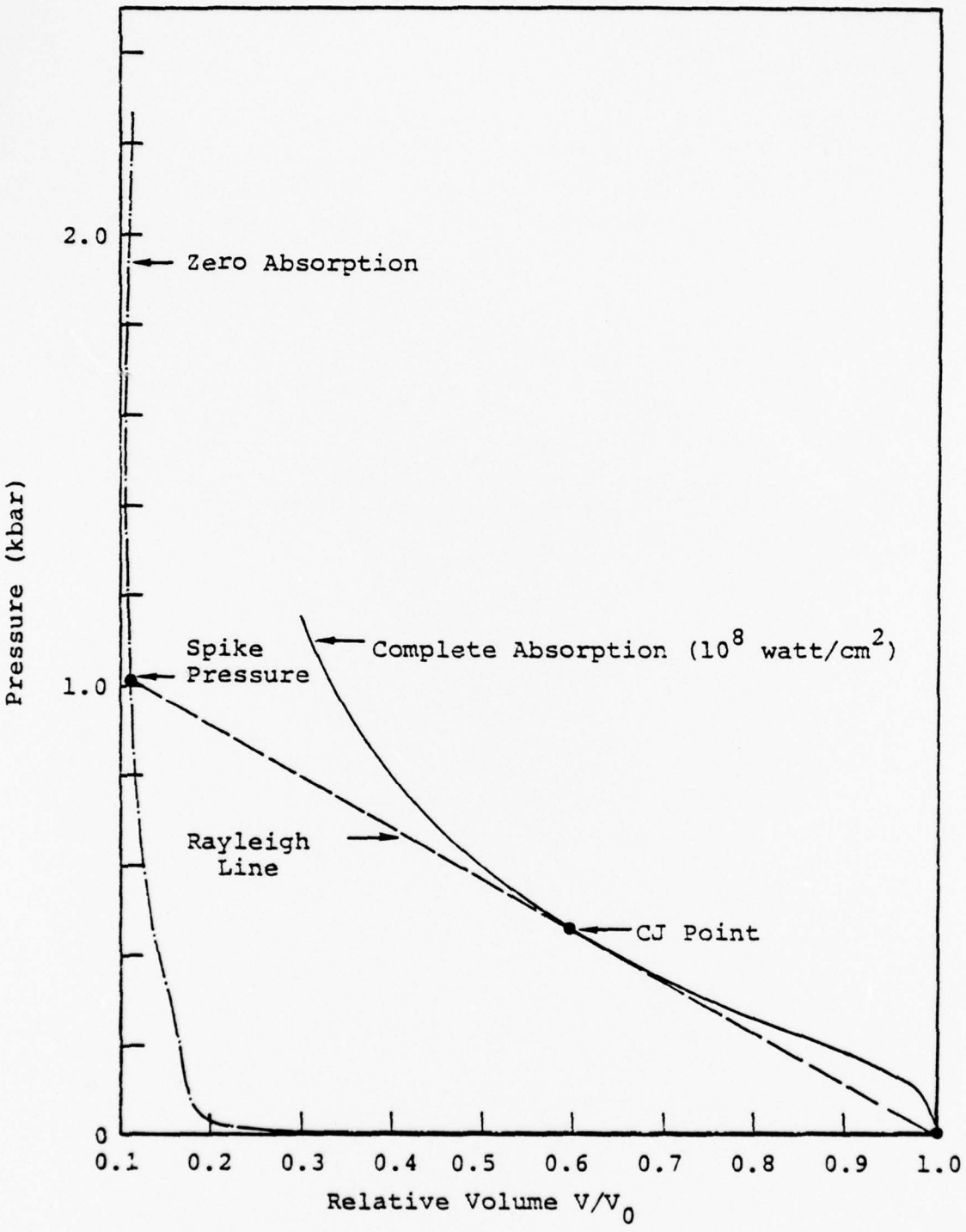


Figure 2. Hugoniots for an ionizable ideal gas corresponding to zero energy absorption and to complete energy absorption for a laser beam flux of 10^8 watt/cm^2.

state absorption wave connecting an initial state P_o, V_o, E_o to a final state P, V, E (Figure 3) is

$$E - E_o = \frac{1}{2}(P + P_o)(V_o - V) + \frac{G}{\rho_o D} , \quad (4)$$

where G is the laser flux (erg/cm^2/sec) and D is the wave velocity, which for a steady-state wave is related to the pressure and volume by

$$D = V_o \sqrt{\frac{P - P_o}{V_o - V}} . \quad (5)$$

The curve labeled "Complete Absorption" in Figure 2 is the Hugoniot as calculated from Equation (4), the P, V, E equation of state as described in Section 2.1, and a laser flux G of 10^8 watt/cm^2. The initial state is $P_o = 10^6$ dyne/cm^2, $T_o = 298.16$ kelvins, $V_o = 885.4$ cm^3/g, and $E_o = 2.213 \times 10^9$ erg/g. The parameters used in the equation of state are:

$f = 5$

$R = 2.9694 \times 10^6$ erg/g/K

$E_o = 14.4$ ev

$A = 4.2662 \times 10^{-7}$ g cm^{-3} K$^{-3/2}$

The CJ point is that point at which the Rayleigh line is tangent to the Hugoniot curve, as indicated in Figure 2. The spike pressure is given by the intersection of this same Rayleigh line with the Hugoniot corresponding to zero energy absorption (calculated from Equation (4) with G = 0). For this equation of state and a flux of 10^8 watt/cm^2, the spike pressure is 1.0 kbar, the CJ pressure is 0.45 kbar, and the wave speed is 10.0×10^5 cm/sec.

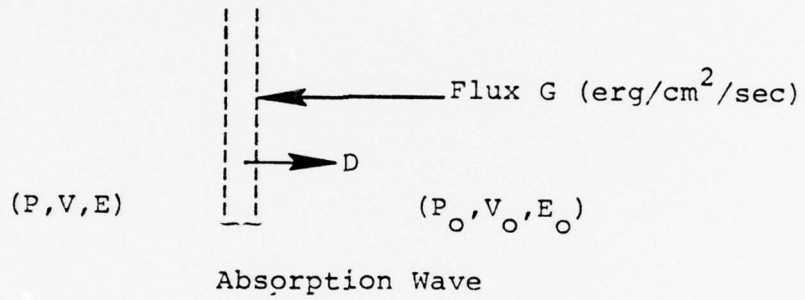


Figure 3. Steady-state absorption wave connecting initial state P_o, V_o, E_o to find state P, V, E.

2.3 NUMERICAL METHOD

The Fortran listings of the one-dimensional FCT modules as given by Boris,[1] modified where necessary to allow for running on the CDC-7600 and Univac 1108 computers, have been incorporated into a preliminary two-dimensional test code. A complete listing is given in Appendix I. The subroutine UPDATE is a realization of the two-dimensional time-splitting procedure as outlined in Reference [2]. The calculations reported here were done using the one-dimensional option, which is capable of operating in either the Eulerian or the Lagrangian mode.

The flux-corrected transport method has been discussed in a series of papers.[1-5] Briefly, it solves the generalized continuity equation of the form

$$\frac{\partial \phi}{\partial t} + \frac{1}{r^m} \frac{\partial}{\partial r} (r^m \phi u) = \text{Source Terms} , \tag{6}$$

where the terms on the left are the one-dimensional equivalent of $(\partial\phi/\partial t) + \nabla \cdot (\phi \bar{u})$. Specifically, in the present case the algorithm is applied in succession to the three equations,

$$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x} (\rho u) = 0, \tag{7}$$

$$\frac{\partial}{\partial t} (\rho u) + \frac{\partial}{\partial x} (\rho u^2) = - \frac{\partial P}{\partial x} , \tag{8}$$

$$\frac{\partial \varepsilon}{\partial t} + \frac{\partial}{\partial x} (\varepsilon u) = - \frac{\partial}{\partial x} (Pu), \tag{9}$$

where

$$\varepsilon \equiv \rho \left(E + \frac{1}{2} u^2 \right) . \tag{10}$$

An alternative form of the last equation is

$$\frac{\partial}{\partial t} (\rho E) + \frac{\partial}{\partial x} (\rho E u) = - P \frac{\partial u}{\partial x} . \qquad (11)$$

In the above equations, which represent the conservation of mass, momentum and energy, respectively, ρ is the density, u is the material velocity, P is the pressure, x is the Eulerian space coordinate, and E is the specific internal energy (erg/g).

2.3.1 Absorption Coefficient

The laser beam intensity as a function of position for one-dimensional propagation in an absorbing medium is given by

$$I = I_o \exp \left[- \left| \int_{x_o}^{x} k_\nu \, dx \right| \right] .$$

The resulting expression used to compute the increase in specific internal energy within a computational zone in time Δt is

$$\rho_m \, \Delta E_m = I_{m+1} \left[1 - \exp \left(- k_\nu \, |\Delta x| \right) \right] \frac{\Delta t}{\Delta x} \text{ erg/cm}^3, \qquad (12)$$

where I_{m+1} is the intensity (erg/cm^2/sec) incident on the right interface of the zone. The absorption coefficient k_ν for a laser frequency ν is given by the sum of the absorption coefficients for free-free transitions in the ionic and atomic fields,

$$k = k_{\nu,i} + k_{\nu,a} , \qquad (13)$$

which are given by the formulae[7]

$$k_{\nu,i} = B_1 Z^2 \lambda^3 x^2 T^{-1/2} (\rho^2/M^2) x [1 - \exp(- h\nu/kT)] \quad (14)$$

$$k_{\nu,a} = B_2 \sigma \lambda^3 x(1-x) T^{3/2} (\rho^2/M^2) [1 - \exp(- h\nu/kT)] \quad (15)$$

where Z is the degree of ionization (equal to 1.0 for these calculations), λ is the laser wavelength in μm, x is the fraction of ionization from Equation (2), M is the gram molecular weight, and σ is the collisional cross-section between electrons and molecules, which was assigned the value 10^{-15} cm^2. The expressions for the numerical coefficients in the above expressions are

$$B_1 = \frac{4}{3} \left(\frac{2\pi}{3m_e k}\right)^{1/2} \frac{e^6 N_{av}^2}{m_e c^4} \times 10^{-12}$$

$$= 4.97 \times 10^{12} \text{ microns}^3 \text{ K}^{1/2} \text{ cm}^5 \text{ mole}^{-2}$$

$$B_2 = \left(\frac{2k}{\pi m_e}\right)^{3/2} \frac{e^2 N_{av}^2}{hc^4} \times 10^{-12}$$

$$= 4.69 \times 10^{17} \text{ micron}^{-3} \text{ K}^{-3/2} \text{ cm}^3 \text{ mole}^{-2}$$

Actually, the precise form of the density and temperature dependence of the absorption coefficient was of little consequence for these calculations, since the detailed structure within the absorption wave was not of interest. For example, at the CJ point corresponding to a flux of 10^8 watt/cm^2, the specific volume is 524 cm^3/gm, the temperature is 41,500 kelvins, and the absorption coefficient as calculated from Equations (14) and (15) for a 10.6 micron laser wavelength is 4100 cm^{-1}, i.e., the absorption front is of the order of 10^{-4} cm thick. Thus, extremely fine zoning

is required to resolve the structure of the absorption front itself.

2.3.2 Test Calculations

The ability of the FCT method to describe the development and propagation of an LSD wave using a zoning which is coarse with respect to the physical thickness of the absorption front is shown in Figure 4. The calculational configuration consists of 200 zones, each 0.01 cm thick, with a rigid wall for the left boundary. The wave was "ignited" by an initial heating of the first three computational zones. An initial energy in these zones of 3.96×10^{11} erg/g at the ambient density of 1.1295×10^{-3} g/cm^3 results in an initial pressure of 96.8 bars, and an initial temperature of 20,360 kelvins, which produces sufficient initial absorption for wave development. The energy in this calculation was updated with Equation (9).

Since the zone width of 0.01 cm is approximately 40 times the absorption length at the CJ point, all of the beam energy is absorbed in one computational zone, and, as expected, such details as the 1.0 kbar spike pressure are not reproduced by the calculation. The other features of the wave profile are reproduced quite well, however. As shown in Figure 4, the wave profiles behind the "pseudo" spike correlate very well with the analytical value of 0.45 kbar at the CJ point and with the analytical values of the slopes at the CJ point as obtained with a simple rarefaction wave centered at x = 0:

$$\left(\frac{\partial P}{\partial x}\right)_{CJ} = \frac{\rho c^2}{\left[1 + \frac{\rho}{2}\frac{d(c^2)}{dP}\right] x} \tag{16}$$

where the square of the sound speed, c^2, and the density, ρ,

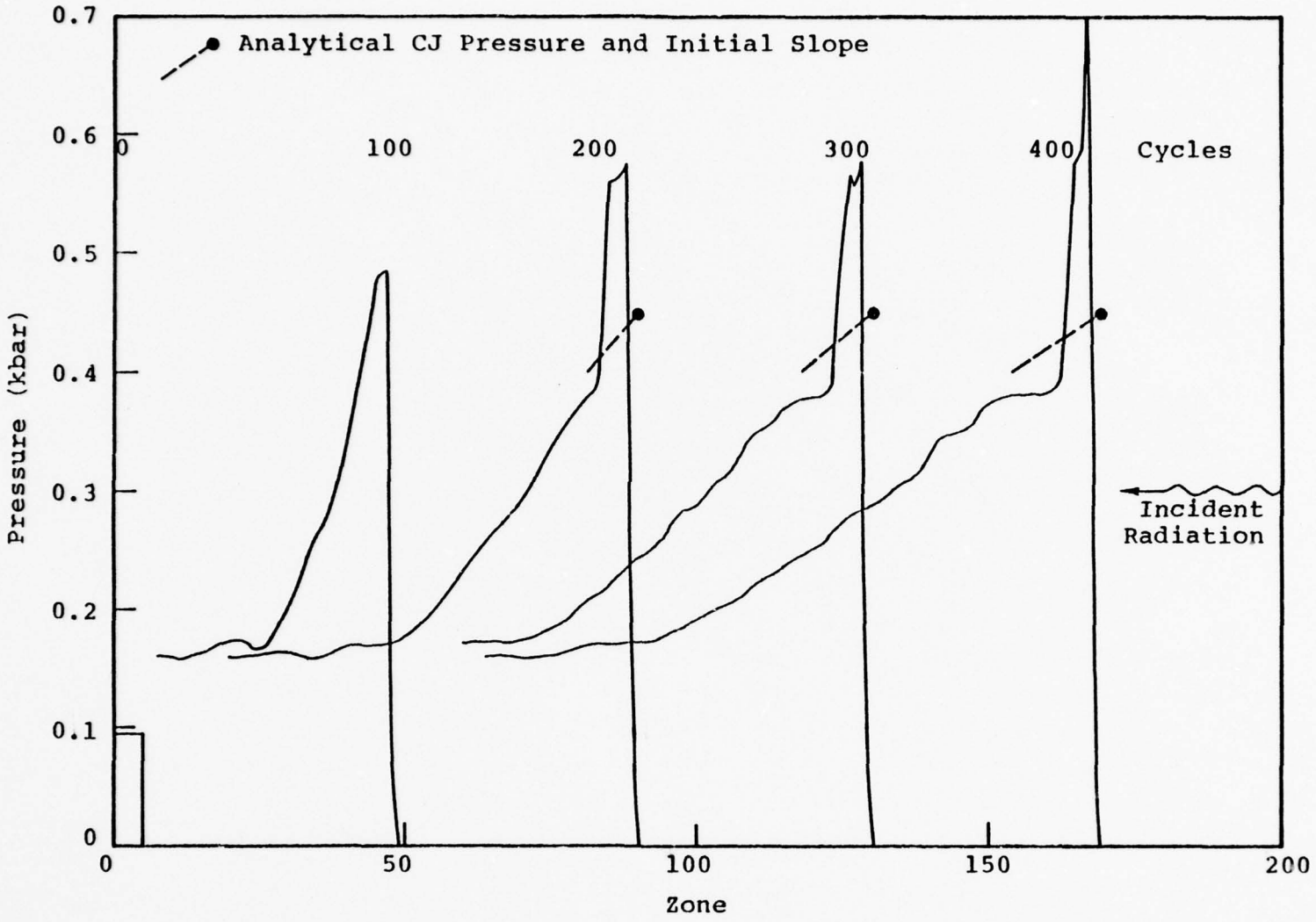


Figure 4. Eulerian FCT calculation of a LSD wave for a flux of 10^8 watt/cm^2. The zone width of 0.01 cm is coarse with respect to the absorption length.

are the values at the CJ point,

$$c^2 = 3.56 \times 10^{11} \text{ cm}^2/\text{sec}^2$$

$$= 1.91 \times 10^{-3} \text{ g/cm}^3$$

$$\frac{d(c^2)}{dP} = 271 \text{ } (\text{cm}^2/\text{sec}^2)/(\text{dyne}/\text{cm}^2) \text{ .}$$

In the interval between 100 cycles (0.466 μsec) and 400 cycles (1.668 μsec), the peak wave pressure progresses from zone 47 to zone 167. The corresponding average wave speed D is then 9.98×10^5 cm/sec, in agreement with the analytical value of 10.0×10^5 cm/sec. Finally, the steady-state pressure at the left boundary is about 0.16 kbar, also in agreement with the analytic value of $\sim$ 0.159 kbar. The conclusion is that the FCT method can propagate an LSD wave properly even with zoning which is much too coarse to resolve the structure within the actual absorption front.

The results of a calculation identical to that described above except that the zone size has been decreased from 0.01 cm to 5.0×10^{-5} cm are given in Figure 5. For this calculation the energy absorption front is about five zones thick, and as a result, the calculated spike pressure of $\sim$ 0.9 kbar is more nearly equal to the analytic value of 1.0 kbar. The ripples superimposed on the rarefaction following the spike are not physical, but are an example of a nonlinear numerical error called "terracing". This type of error, which occurs on the flanks of steep gradients, is discussed in Reference [2].

The two calculations described above were done using the FCT modules in the Eulerian mode, i.e., the zone boundaries were fixed in space. The results of a calculation using the Lagrangian mode are shown in Figure 6. The initial

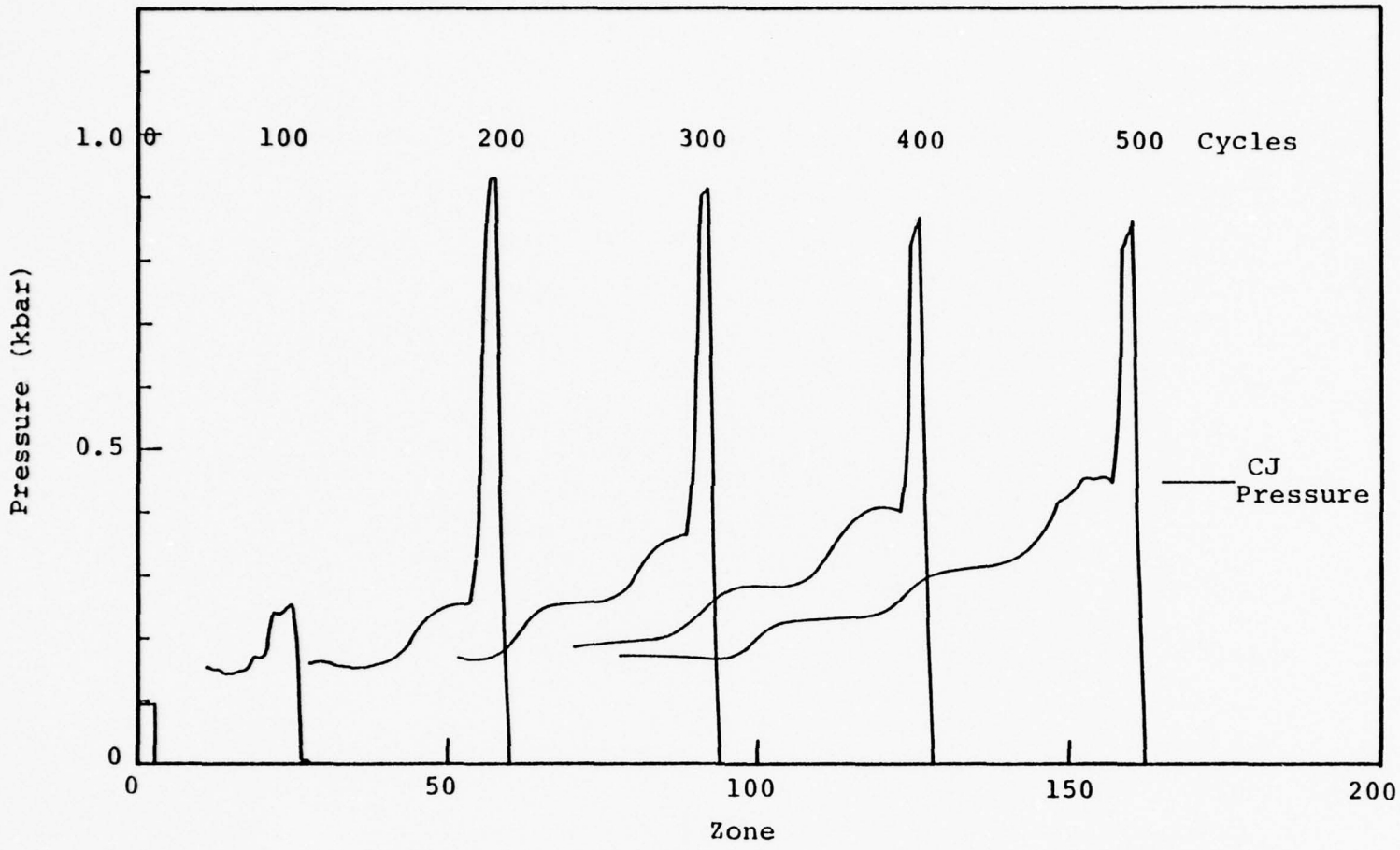


Figure 5. Eulerian FCT calculation of a LSD wave for a flux of 10^8 watt/cm^2. The zone width of 5.0×10^{-5} cm is of the order of the absorption length.

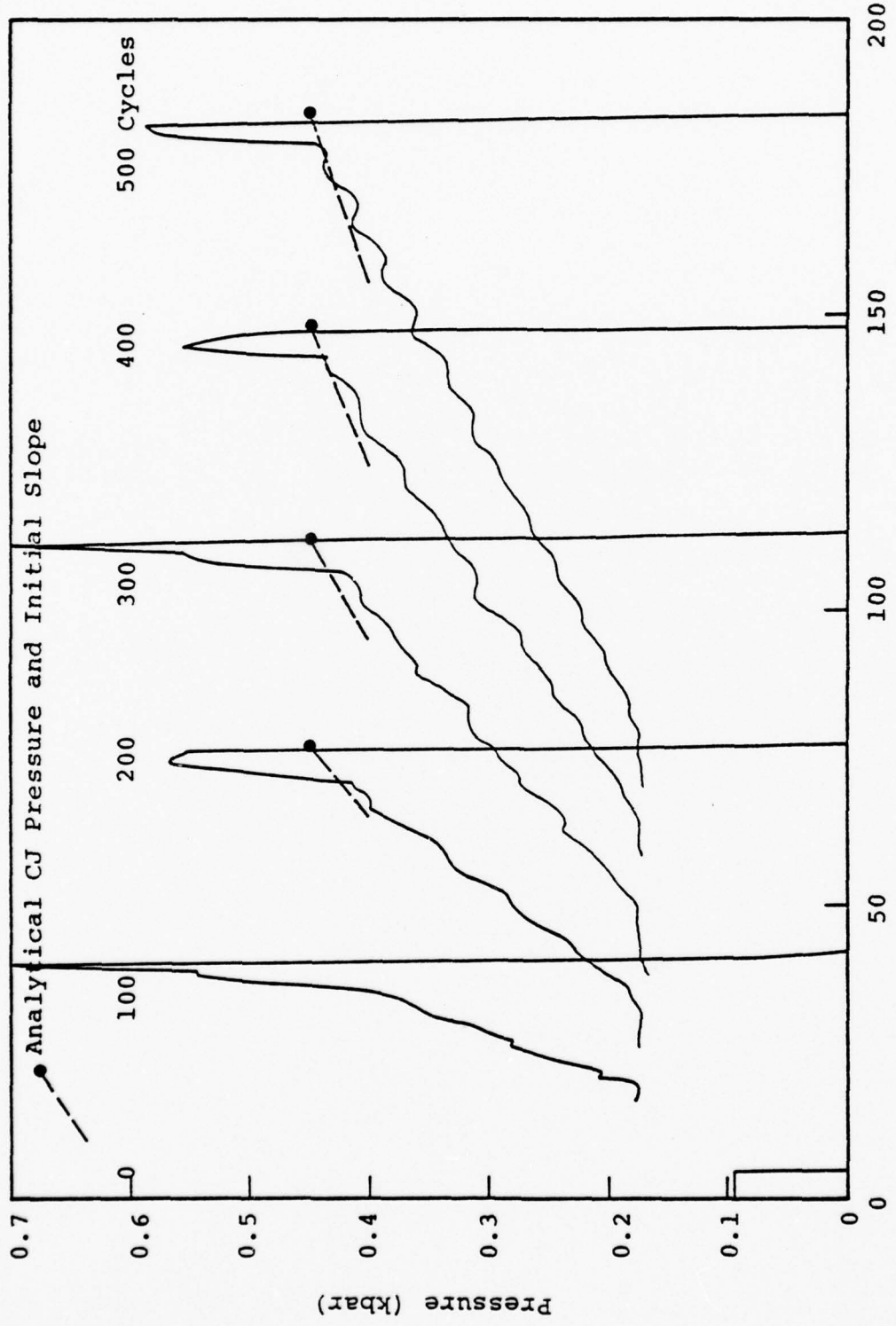


Figure 6. Lagrangian FCT calculation of a LSD wave for a flux of 10^8 watt/cm^2. The initial zone width of 0.01 cm is coarse with respect to the absorption length.

zone width was 0.01 cm, as in the first calculation, but the zone boundaries were allowed to move with the material velocity at each point. This calculation also reproduced the main features of the wave quite well, even though the initial zone width was very large compared with the absorption length. The average wave speed from cycle 100 (0.3999 μsec) to cycle 500 (1.8182 μsec) was 10.0×10^5 cm/sec, in agreement with the analytic value of 10.0×10^5 cm/sec for a flux of 10^8 watt/cm^2. The wave profiles also correlate well with the pressure and the initial slope at the CJ point. It will be noticed from Figure 6 that the peak pressure of the pseudo spike fluctuates considerably as the wave progresses. This fluctuation is a result of the coarse zoning, but it is not an instability; it arises because at times the conditions at the shock front are such that the energy absorption occurs across two computational zones instead of one. When this happens, the pressure in the first zone approaches more nearly the analytical spike pressure.

The results of a calculation to check the use of an artificially long absorption length are shown in Figure 7. The zone width for this Eulerian calculation was 0.01 cm; but if the value of k_ν as calculated from Equations (14) and (15) was such that $k_\nu |\Delta x| > 1.0$, it was reset to 1.0 for that zone. In other words, the beam intensity was not allowed to decrease by more than a factor of e^{-1} across any computational zone. The result of this numerical limiting is an absorption-front thickness of four or five computational zones; and, as expected, the peak spike pressure is close to the analytical value of 1.0 kbar for a flux of 10^8 watt/cm^2. A disadvantage of artificially limiting the absorption coefficient is that the time required to attain steady state is unrealistically long. From Figure 7 it is seen that even after 400 computational cycles, or a travel of 1.4 cm, the pressure behind the spike has still not reached the steady-state CJ value of 0.45 kbar.

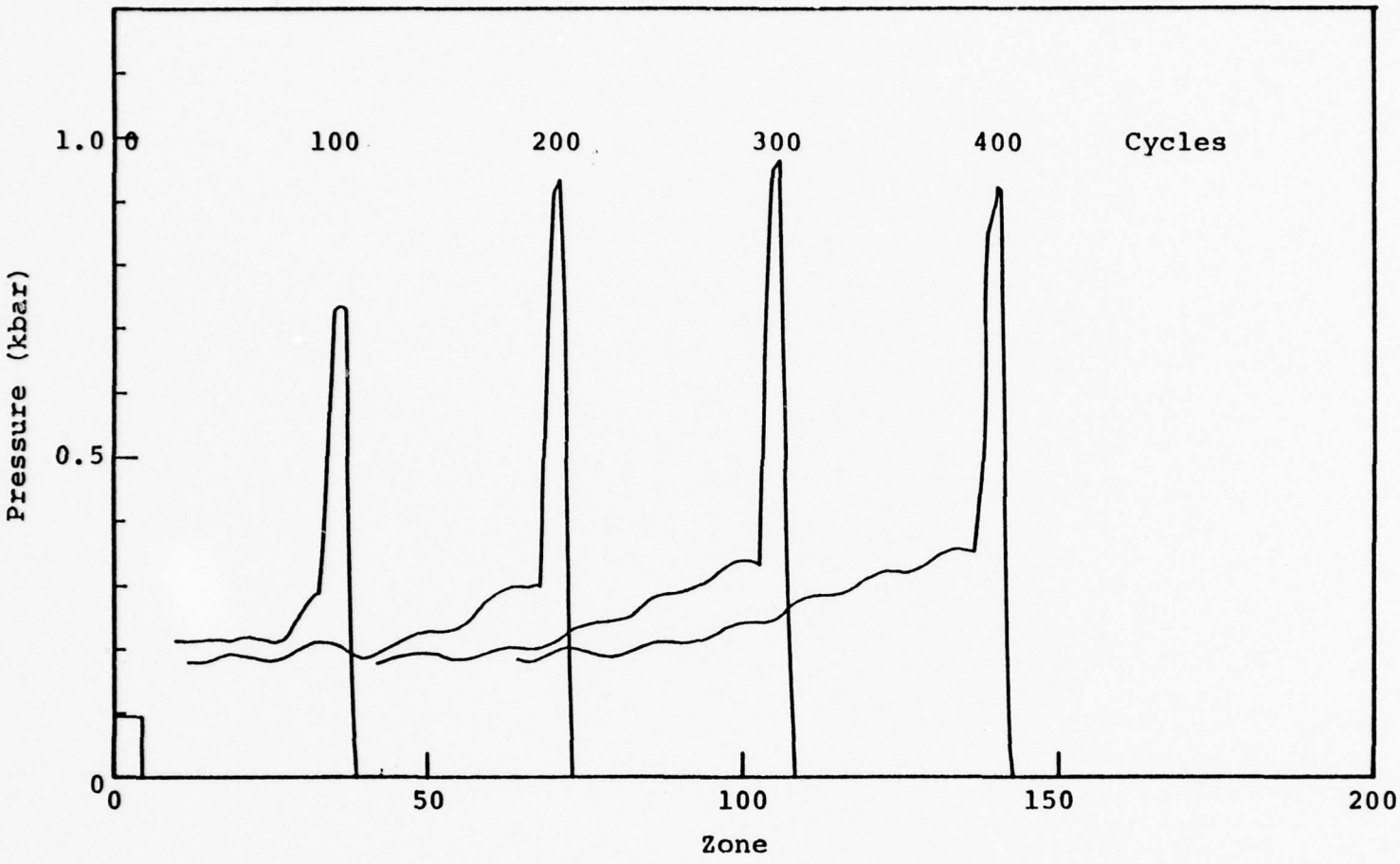


Figure 7. Eulerian FCT calculation of a LSD wave for a flux of 10^8 watt/cm^2. The absorption length has been artificially forced to be of the same order as the zone width of 0.01 cm.

2.4 CONCLUSIONS

The FCT method is a convenient and satisfactory numerical procedure for describing the propagation of one-dimensional LSD waves. In particular, the method yields satisfactory results even for zoning which is much too coarse to resolve the structure within the energy absorption front.

3. LASER BEAM PROPAGATION IN GASES AND PLASMAS

The propagation of a laser beam is a highly specialized problem from the standpoint of radiation transport in general, since the radiation field is very sharply peaked with respect to both angular distribution and frequency distribution. These characteristics can only be maintained in a material medium whose properties vary continuously and slowly over finite intervals of space and time, so that the variations may be considered as perturbations within these intervals. At the boundaries of such intervals, discontinuous material properties may be taken into account in the beam model by introducing additional beams; such may also originate in the interior of an interval by means of stimulated Raman scattering and similar processes. In this section the propagation of a single beam within a continuous medium will be discussed. The properties of the medium will be assumed given, although they may depend in various ways upon the radiation field itself.

For a beam with axis in the z direction in a medium with complex refractive index, n, the amplitude E satisfies the scalar wave equation

$$\nabla^2 E + \frac{\partial^2 E}{\partial z^2} - \frac{n^2}{c^2}\frac{\partial^2 E}{\partial t^2} = 0 \qquad (17)$$

where the ∇ operator is restricted to the x-y plane. The perturbation formulation discussed above suggests that the amplitude be written as the product of a slowly-varying envelope and a fast unperturbed part:

$$E(x,y,z,t) = \psi(x,y,z,t)e^{i(k_o z - \omega_o t)} \qquad (18)$$

where ω_o is the central laser beam frequency and $k_o = n_o\omega_o/c$,

with n_o being a constant real index of refraction assumed as a base of reference. Let

$$\alpha = 2 \frac{\omega_o}{c} \text{Im } n = \text{Im } k \tag{19}$$

$$\mu = \frac{\text{Re } n - n_o}{n_o} = \frac{\text{Re } k - k_o}{k_o} \tag{20}$$

represent the local absorption coefficient and index perturbation, respectively, then on dropping terms of second order in $\partial/\partial z$, $\partial/\partial t$, α, μ, and $\omega-\omega_o$ there results the time-dependent paraxial wave equation

$$\nabla^2\psi + 2ik_o \left(\frac{\partial\psi}{\partial z} + \frac{1}{v}\frac{\partial\psi}{\partial t}\right) + \left(ik_o\alpha + 2k_o^2\mu\right)\psi = 0 \tag{21}$$

where v is the group velocity. Fleck[8] solved this equation directly for the real and imaginary parts of ψ in the time-independent, axially symmetric case. Glass[6] followed the alternative approach of separating ψ into amplitude and phase parts. The latter method leads, as will be shown, to a formulation in terms of variables which have simple geometrical interpretations and are well suited to calculations with adaptive coordinate grids. We have adopted this latter approach in the present effort.

Following Reference [6] we introduce real intensity and inclination variables I and $\underline{u}$ by the relations

$$\psi = \sqrt{I}\, e^{i\phi} \tag{22}$$

$$\underline{u} = \frac{1}{k_o} \nabla\phi \tag{23}$$

With these definitions, the real and imaginary parts of (21) yield the pair of equations

$$\left(\frac{\partial}{\partial z} + \frac{1}{v}\frac{\partial}{\partial t} + \underline{u} \cdot \nabla\right) I = -(\alpha + \nabla \cdot u)\, I \tag{24}$$

$$\left(\frac{\partial}{\partial z} + \frac{1}{v}\frac{\partial}{\partial t} + \underline{u} \cdot \nabla\right) \underline{u} = \nabla(\mu + P) \tag{25}$$

where the diffraction potential P is defined as

$$P = \frac{1}{4k_o^2}\left(\frac{\nabla^2 I}{I} - \frac{1}{2}\left(\frac{\nabla I}{I}\right)^2\right) \tag{26}$$

These equations may be integrated along characteristics, or trajectories, which are the analogs in Fresnel diffraction theory of the rays of geometric optics. These trajectories are defined by the pair of equations

$$\frac{d}{dz}\underline{r} = \underline{u} \tag{27}$$

$$\frac{d}{dz}t = \frac{1}{v} \tag{28}$$

where $\underline{r}$ is the radius vector projection in the plane normal to the z axis and t is the time. Along such trajectories (24) and (25) take the form

$$\frac{d}{dz} I = -(\alpha + \nabla \cdot \underline{u})\, I \tag{29}$$

$$\frac{d}{dz}\underline{u} = \nabla(\mu + P) \tag{30}$$

Let us consider an arbitrary portion of the beam cross-section for given z, t. This is a region R of the xy plane, bounded by a set of points $\underline{r}$ at which (27) holds. On integrating (24) over this region, with boundary terms defined by (27), one finds

$$\iint_R \left(\alpha I + \frac{1}{v}\frac{\partial I}{\partial t}\right) dx\, dy + \frac{d}{dz}\iint_R I\, dx\, dy = 0 \qquad (31)$$

which implies that energy flows only along, not across, trajectories. We may, in fact, replace (29) by (31) in order to have an explicit equation for energy conservation. Similarly, with the aid of (27) we may replace (30) by

$$\frac{d^2}{dz^2}\underline{r} = \nabla(\mu + P) \qquad (32)$$

The time-dependence represented by (28) is a result of the finiteness of the group velocity. It requires an interpolation procedure, since the differentiations in the x and y directions require constancy of t as well as z, and unless the group velocity is constant, trajectories that are simultaneous at one value of z will not be simultaneous at other values. Time-dependence may be of considerable importance in some applications, but there are many in which the scale of the problem is such that finite propagation velocity effects are not of interest. When this is the case, we may set $v \to \infty$ in the above equations, and drop (28). Trajectories are then reinterpreted as space paths rather than space-time paths. If the material properties and boundary conditions are time-dependent, the trajectories will, of course, change with time, but no time derivatives will appear in the radiation propagation equations.

A second simplification which is worth introducing is the restriction to axial symmetry. This is particularly convenient in a developmental program, since it greatly reduces the number of trajectories required for a given spatial resolution. The time-independent axially symmetric forms of (31) and (32) may then be written

$$\frac{dW_i}{dz} + \alpha_i(z)\ W_i = 0 \tag{33}$$

$$\frac{d^2 r_i}{dz^2} = \frac{\partial}{\partial r}\ (\mu + P) \tag{34}$$

where

$$W_i(z) = \int_{r_i}^{r_{i+1}} 2\pi r I(r,z)\ dr \tag{35}$$

is the beam power in the ring $r_i < r < r_{i+1}$, and

$$\alpha_i(z) = \int_{r_i}^{r_{i+1}} 2\pi r \alpha(r,z)\ I(r,z)\ dr / W_i(z) \tag{36}$$

is the effective absorption coefficient in the ring.

It may be noted that (33) does not involve the radial variable r at all, and in (34) r appears as a dependent variable. This suggests that the right side of (34) be evaluated in terms of an independent variable which is related to W. It is also important to account properly for the radial boundary conditions, both at r = 0 and at the outer boundary. Fleck[8] devised a method of cylindrical cubic spline representation specifically for the r variable. An alternative approach becomes possible with other variable choices, which may provide certain advantages.

For the evaluation of the right side of (34) we choose as dependent variable the cumulative area to radius r (per radian)

$$y = \frac{1}{2}\ r^2 \tag{37}$$

and as independent variable

$$x = - \log (1-S(r)/S_T) \tag{38}$$

where

$$S(r) = \int_0^r 2\pi r I(r)\, dr \tag{39}$$

is the beam power within radius r, and $S_T = S(\infty)$ is the total power. It is clear that a Gaussian beam profile

$$I(r) = \frac{2S_T}{\pi w^2} \exp (-2r^2/w^2) \tag{40}$$

is represented in these variables as the straight line

$$y = \frac{1}{4} w^2 x \tag{41}$$

Since in the absence of perturbations the Gaussian profile propagates without change of shape (except for changes in the scale radius w), it is natural to assume a Gaussian shape for the beam profile at the outer edge of the calculational mesh as a boundary condition which is not likely to propagate spurious information into the interior. Such a boundary condition, in view of (41), is easily imposed with the variable definitions chosen.

The derivative terms appearing in (26) and on the right side of (34) are easily evaluated. With derivatives by x denoted by dots, one finds from (37), (38) and (39)

$$I = S_T\, e^{-x/\dot{y}} \tag{42}$$

$$T_1 = \frac{1}{rI}\frac{dI}{dr} = \frac{1}{I}\frac{dI}{dy} = -\left(\frac{1}{\dot{y}} + \frac{\ddot{y}}{\dot{y}^2}\right) \tag{43}$$

$$T_2 = \frac{d}{dy}T_1 = (\dddot{y} - \ddot{y} + 2\ddot{y}^2/\dot{y})/\dot{y}^3 \tag{44}$$

$$T_3 = \frac{d}{dy}T_2 = (\ddddot{y} - \dddot{y})/\dot{y}^4 + (7\ddot{y}\,\dddot{y} - 3\ddot{y}^2)/\dot{y}^5 - 8\ddot{y}^3/\dot{y}^6 \tag{45}$$

Then (26) may be written

$$P = \frac{1}{4k_o^2}\left(\frac{1}{Ir}\frac{d}{dr}\; r\,\frac{dI}{dr} - \frac{1}{2}\left(\frac{1}{I}\frac{dI}{dr}\right)^2\right)$$

$$= \frac{1}{4k_o^2}\left(\frac{1}{I}\frac{d}{dy}\,(2yI\,T_1) - yT_1^2\right)$$

$$= \frac{1}{4k_o^2}\left(2T_1 + yT_1^2 + 2yT_2\right) \tag{46}$$

so that

$$\frac{\partial P}{\partial r} = \frac{r}{4k_o^2}\left(4T_2 + T_1^2 + 2yT_1T_2 + 2yT_3\right). \tag{47}$$

The refraction term in (34) is simply

$$\frac{\partial \mu}{\partial r} = r\dot{\mu}/\dot{y} \tag{48}$$

It may be noted that the right sides of (47), (48) and therefore of (34) vanish properly at r = 0.

The derivatives of y and μ with respect to x may be evaluated from spline representations. In order to construct these, trajectory values of x, y are first set up. These are determined by (37) and (38) following each integration

step for (33) and (34):

$$x_j = -\log\,(1 - S_j/S_T),\ j = 1,\ n \tag{49}$$

where

$$S_1 = 0 \tag{50}$$

$$S_{j+1} = \sum_{i=1}^{j} W_i,\ j = 1,\ n \tag{51}$$

$$S_T = S_{n+1} \tag{52}$$

$$y_i = \frac{1}{2}\, r_j^2 \tag{53}$$

According to (45) derivatives of y up to fourth order are required. A quintic spline representation provides continuous derivatives through fourth order. A recently developed natural quintic spline code, QUINAT[9] was therefore modified for the purpose. The procedure consists of two steps:

1. After an integration step for the W_j, Equation (33), and calculation of new values of x_j, Equation (49), the first part of the spline calculation is done. This consists of setting up a matrix of coefficients and decomposing it into triangular and diagonal factors, using a Cholesky method. The results of this step depend only on the x_j and can be used repeatedly for calculation of spline representation of other trajectory variables in step (2).

2. The second step is done for each new or modified set of trajectory values of α, μ and y. It yields coefficients B_j, C_j, D_j, E_j, and F_j for the natural spline polynomials

$$A(x) = A_j + B_j\,(x-x_j) + C_j\,(x-x_j)^3$$

$$+ E_j\,(x-x_j)^4 + F_j\,(x-x_j)^5,\ x_j \leq x < x_{j+1},$$

$$j = 1,\ n \qquad (54)$$

where the A_j are the prescribed values of some functions at the points x_j. The representation A(x) and its first four derivatives are continuous at the interior nodes x_j, j = 2, n - 1. At the end-points x_1 and x_n the third, fourth and fifth derivatives of A vanish automatically.

In the original QUINAT program[9] it was not possible to separate these two steps. By explicitly factorizing the coefficient matrix in step (1) this separation is made possible. Since the operations in step (2) must be carried out many times compared to the more time-consuming ones in step (1), the gain in efficiency is large.

It remains to specify integration methods for (33) and (34). The coefficients α_j defined by (33) are evaluated from the spline representations of $\alpha(x)$ and y(x). With I(x) defined by (42) the integration in (36) can be done analytically. If the α_i vary slowly with z, they may be taken as constant within each increment of the z variable, in which case (33) may be integrated directly:

$$W_j(z + \Delta z) = W_i(z)\ \exp\ (-\ \alpha_j\ \Delta z) \qquad (55)$$

For x_j strongly dependent on z it may be preferable to integrate the equivalent form

$$\frac{d}{dz} \log W_i = -\ \alpha(z) \qquad (56)$$

using a higher-order numerical procedure.

For integration of Equation (34) a predictor-corrector method is used. The method chosen is a generalization to nonuniform meshes of Störmer's method for solving second-order equations of the form

$$\frac{d^2r}{dz^2} = G(r,z) \ . \tag{57}$$

Specifically, let

$$h = z_{n+1} - z_n$$

$$g = z_n - z_{n-1}$$

The corrector, or closed, formula is then

$$gr_{n+1} = (h + g)\ r_n - h\ r_{n-1} + \frac{1}{12}\ h\ (g^2 + gh - h^2)\ r''_{n-1}$$

$$+ \frac{1}{12}\ (g + h)\ (g^2 + 3gh + h^2)\ r''_n + \frac{1}{12}\ g\ (h^2 + gh \tag{58}$$

$$- g^2)\ r''_{n+1} + \frac{1}{360}\ hg\ (g^2 - h^2)\ (2g^2 + 5gh + 2h^2) r^v(z_n + \varepsilon)$$

where the last term indicates the order of error incurred by dropping it. The predictor, or open formula chosen is

$$gr_{n+1} = (h + g)r_n - hr_{n-1} + \frac{1}{6}\ h\ (g^2 - h^2)r''_{n-1}$$

$$+ \frac{1}{6}\ h\ (h + g)\ (h + 2g)\ r''_n + \frac{1}{24}\ (3g^4\ h - 2h^3g^2$$

$$+ h^4g)\ r^{iv}\ (z_n + \varepsilon) \tag{59}$$

Following an initial estimate of r_{n+1} by (59), the right side of (34) is evaluated as described above to provide an estimate of r''_{n+1}. The process is then iterated to convergence using the corrector (58).

The initialization of the trajectories does not present any particular problem. It is useful to record some relations characterizing Gaussian beams, in which the intensity varies as in (40) or (41). The scale radius w of such a beam has a z-dependence given by the hyperbola[10]

$$w(z)^2 = w_o^2 + \frac{4}{k_o^2 w_o^2} (z - z_o)^2 \tag{60}$$

where z_o, w_o are the values of z and w at the waist of the beam. Each trajectory is a similar hyperbola

$$r_j(z) = \frac{r_j(z_o)}{w_o} w(z) \tag{61}$$

so that

$$\frac{d^2}{dz^2} r_j = \frac{4 r_j}{k_o^2 w(z)^4} \tag{62}$$

The result (62) is also deriverable from (34), (41) and (47) with the added assumptions $\alpha = \mu = 0$. The values of r and r" needed to initialize trajectories in a Gaussian beam may be determined very simply by these relations.

Material properties k_o, $\alpha(r,z)$ and $\mu(r,z)$ may be specified in a variety of ways. A dispersion relation for partially ionized gas,

$$k^2 = k_o^2 - \frac{\omega_p^2}{c^2} \frac{\omega}{\omega + i(\nu_i + \nu_n)} \tag{63}$$

where ω_p is the electron plasma frequency and ν_i, ν_n are the electron collision frequencies with ions and neutrals respectively, is being used in test calculations. The

quantities α and μ are obtained from (63) by extracting the complex square root and applying (19) and (20).

A Fortran listing of the routines which perform the operations discussed above is presented in Appendix II.

REFERENCES

1. Boris, J. P., "Flux-Corrected Transport Modules for Solving Generalized Continuity Equations," NRL Memorandum Report 3237, March, 1976.

2. Book, D. L., J. P. Boris, and K. Hain, "Generalizations of the Flux Corrected Transport Technique," NRL Memorandum Report, 3021, April, 1975.

3. Boris, J. P. and D. L. Book, "Flux-Corrected Transport I: SHASTA, A Fluid Algorithm That Works," J. Comp. Phys. II, 38-69, 1973.

4. Book, D. L., J. P. Boris, and K. Hain, "Flux-Corrected Transport II: Generalizations of the Method," J. Comp. Phys. 18, 248-283, 1975.

5. Boris, J. P. and D. L. Book, "Flux-Corrected Transport III: Minimal-Error FCT Algorithms," J. Comp. Phys. 20, 397-431, 1976.

6. Glass. A., Laser-Fusion Program Semi-Annual Report, UCRL-50021-73-1, p. 143-145, Jan-June, 1973.

7. Zel'dovich, Ya. B. and Yu. P. Raizer, Physics of Shock Waves and High-Temperature Hydrodynamic Phenomena, Volume I, Academic Press, (p. 259), 1966.

8. Fleck, J. A. Jr., A cubic Spline Method for Solving the Wave Equation of Nonlinear Optics, J. Comp. Phys. 16, 324-341, 1974.

9. Herriot, J. G. and C. H. Reinsch, Algorithm 507, Procedures for Quintic Natural Spline Interpolation [E1], Collected Algorithms from ACM, or ACM Transactions on Mathematical Software 2, p. 281-289, 1976.

10. Kogelnik, H. and T. Li, Laser Beams and Resonators, Appl. Opt. 5, p. 1550-1567, 1966.

APPENDIX I

LISTING OF LSD WAVE PROPAGATION ROUTINES

```
***** MAINC *****

      1.                PROGRAM MAINC(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)
      2.          C
      3.          C     TWO-D DRIVER FOR ETBFCT
      4.          C
      5.                REAL RNEW(202,12),VZNEW(202,12),VRNEW(202,12),ENEW(202,12)
      6.                REAL PNEW(202,12),F(202,12),TOLD(202,12),SNDSPD(202,12)
      7.                REAL FLUX(202,12),RADZ(202),RADR(12),RBC(2,4),LBC(2,4)
      8.                REAL ZERO(202),UNIT(202)
      9.          C
     10.                COMMON/UPDT/ RNEW, VZNEW, VRNEW, ENEW, PNEW, F, TOLD, SNDSPD, FLUX
     11.                COMMON/UPDT/ RADZ, RADR, RADZR, RADZL, RADRR, RADRL, NZ, NR, RBC
     12.                COMMON/UPDT/LBC, ZERO, UNIT, DELTAZ, DELTAR, DELTAT, COUR, ENRGMN
     13.                COMMON/UPDT/FLAG
     14.                COMMON/DELZ/DELZL(202)
     15.          C
     16.                COMMON/EDITC/NREDIT, NDELTR, ISTEPX, LPRINT, TIME
     17.                COMMON/EOSC/ ENRG, RHO, P, T, FRAC, A, B, DEGF, RM, SND
     18.                COMMON/FLXC/ FZERO, XLAMDA, Z, GMWT, CUTOFF
     19.          C
     20.          C     READ EOS CONSTANTS, ETC.
     21.                READ(5,100) XLAMDA,A,B,DEGF,GASCON,GMWT,Z,COUR,CUTOFF
     22.            100 FORMAT(4E20.14)
     23.          C
     24.                READ(5,100) FZERO, RZERO, ENRGL, ENRGR, DELTAZ, DELTAR, ENRGMN
     25.                READ(5,100) FLAG
     26.          C
     27.                READ(5,101) MAXSTP,IPRINT,NZ,NR,LPRINT,MZ,MR,NREDIT,NDELTR
     28.            101 FORMAT(8I10)
     29.          C
     30.                READ(5,103) RBC
     31.                READ(5,103) LBC
     32.            103 FORMAT(8F10.0)
     33.          C
     34.                RM=GASCON
     35.                RINV=1./RZERO
     36.                ELEFT=RZERO*ENRGL
     37.                ERIGHT=RZERO*ENRGR
     38.          C
     39.                RHO=RZERO
     40.                ENRG=ENRGL
     41.                CALL EOS
     42.                TLEFT=T
     43.                PLEFT=P
     44.                FLEFT=FRAC
     45.                SNDL=SND
     46.          C
     47.                ENRG=ENRGR
     48.                CALL EOS
     49.                TRIGHT=T
     50.                PRIGHT=P
     51.                FRIGHT=FRAC
     52.                SNDR=SND
```

```
******  MAINC  ******

C
      VZRIGH=0.
      VZLEFT=0.
      VRRIGH=0.
      VRLEFT=0.
C
      WRITE(6,102) XLAMDA,A,B,DEGF,GASCON,GMWT,Z,COUR,FZERO,RZERO,
     .   ENRGL,ENRGR,DELTAZ,DELTAR,MAXSTP,IPRINT,NZ,NR,LPRINT,MZ,MR,
     .   NREDIT, NDELTR
  102 FORMAT('1'//' XLAMDA=',E14.8/' A=',E14.8/' B=',E14.8/' DEGF=',E14.
     .8/' GASCON=',E14.8/' GMWT=',E14.8/' Z=',E14.8/' COUR=',E14.8/' FZE
     .RO=',E14.8/' RZERO=',E14.8/' ENRGL=',E14.8/' ENRGR=',E14.8/' DELTA
     .Z=',E14.8/' DELTAR=',E14.8/' MAXSTP=',I5/' IPRINT=',I5/' NZ=',I5/'
     . NR=',I5/' LPRINT=',I5/' MZ=',I5/' MR=',I5/' NREDIT=',I5/' NDELTR=
     .',I5)
C
      WRITE(6,104) CUTOFF,RBC,LBC
  104 FORMAT(1X,'CUTOFF=',E14.8/' RBC(1,1)=',F5.0/' RBC(2,1)=',F5.0/' RB
     .C(1,2)=',F5.0/' RBC(2,2)=',F5.0/' RBC(1,3)=',F5.0/' RBC(2,3)=',F5.
     .0/' RBC(1,4)=',F5.0/' RBC(2,4)=',F5.0/' LBC(1,1)=',F5.0/' LBC(2,1)
     .=',F5.0/' LBC(1,2)=',F5.0/' LBC(2,2)=',F5.0/' LBC(1,3)=',F5.0/' LB
     .C(2,3)=',F5.0/' LBC(1,4)=',F5.0/' LBC(2,4)=',F5.0)
      WRITE(6,105) ENRGMN
  105 FORMAT(1X,'ENRGMN=',E14.8)
      WRITE(6,106) FLAG
  106 FORMAT(1X,'FLAG=',F5.0)
C     SET UP THE GRID
C
      DO 10 I=1,NZ
   10 RADZ(I)=FLOAT(I-1)*DELTAZ
      RADZR=RADZ(NZ) + 0.5*DELTAZ
      RADZL=RADZ(1) - 0.5*DELTAZ
      DO 11 J=1,NR
   11 RADR(J)=(FLOAT(J-1) + .5)*DELTAR
      RADRR=RADR(NR) + 0.5*DELTAR
      RADRL=RADR(1) - 0.5*DELTAR
C
C     INITIALIZE THE MESH
      DO 12 J=1,NR
      DO 12 I=1,NZ
      RNEW(I,J)=RZERO
      VZNEW(I,J)=0.
      VRNEW(I,J)=0.
      PNEW(I,J)=PRIGHT
      TOLD(I,J)=TRIGHT
      F(I,J)=FRIGHT
      ENEW(I,J)=ERIGHT
      SNDSPD(I,J)=SNDR
   12 CONTINUE
C
      DO 17 I=1,NZ
   17 DELZL(I)=DELTAZ
      MMM=MAX0(NZ,NR)
      DO 115 I=1,MMM
```

```
****** MAINC ******

                    ZERO(I)=0.
                115 UNIT(I)=1.
              C
              C     FIRST MZ BY MR ZONES ARE HEATED INITIALLY
              C
                    DO 13 J=1,MR
                    DO 13 I=1,MZ
                    ENEW(I,J)=ELEFT
                    PNEW(I,J)=PLEFT
                    TOLD(I,J)=TLEFT
                    F(I,J)=FLEFT
                    SNDSPD(I,J)=SNDL
                 13 CONTINUE
              C
                    TIME=0.
              C
              C     LOOP OVER TIME STEPS
              C
                    DO 9999 ISTEP=1,MAXSTP
              C
                    ISTEPX=ISTEP
              C
                    CALL FLX
              C
                    MXX=MR+1
                    DO 150 J=MXX,NR
                    DO 150 I=1,NZ
                150 FLUX(I,J)=0.
              C
                    CALL TDELT
              C
                    IF(MOD(ISTEP-1,IPRINT).NE.0) GO TO 14
                    CALL EDIT
                 14 CONTINUE
              C
                    CALL UPDATE
              C
                    TIME=TIME + DELTAT
              C
                9999 CONTINUE
              C
                    STOP
                    END
```

```
****** ABSORB ******

      1.
      2.                C
      3.                C     CALCULATES ABSORPTION COEFF FOR FREE ELECTRONS IN IONIC AND
      4.                C         ATOMIC FIELDS
      5                 C
      6.                C     T IS TEMP IN KELCINS
      7.                C     V IS SPEC VOLUME IN CC/GM
      8.                C     XLAMDA IS WAVELENGTH OF INCIDENT RADIATION IN MICRON
      9.                C     Z=1 FOR SINGLE IONIZATION
     10.                C     FRAC IS FRACTION IONIZATION = N-/NZERO
     11.                C     GMWT IS GM MOLECULAR WEIGHT
     12.                C
     13.                      ONE=XLAMDA**3
     14.                      TWO=1. - EXP(-14387.89/(XLAMDA*T))
     15.                      THREE=(V**2)*(GMWT**2)
     16.                      TSQRT=SQRT(T)
     17.                      TERM=ONE*TWO/THREE
     18.                C
     19.                      A1=4.96985E+12*(Z**2)*TERM*(FRAC**2)/TSQRT
     20.                C
     21.                      SIGMA=1.E-15
     22.                C
     23.                      A2=4.68544E+17*T*TSQRT*TERM*FRAC*(1.-FRAC)*SIGMA
     24.                C
     25.                      ABSORB=A1 + A2
     26.                C
     27.                      RETURN
     28.                      END
```

****** EDIT ******

```
      SUBROUTINE EDIT
C
      REAL RNEW(202,12),VZNEW(202,12),VRNEW(202,12),ENEW(202,12)
      REAL PNEW(202,12),F(202,12),TOLD(202,12),SNDSPD(202,12)
      REAL FLUX(202,12),RADZ(202),RADR(12),RBC(2,4),LBC(2,4)
      REAL ZERO(202),UNIT(202)
C
      COMMON/UPDT/ RNEW, VZNEW, VRNEW, ENEW, PNEW, F, TOLD, SNDSPD, FLUX
      COMMON/UPDT/ RADZ, RADR, RADZR, RADZL, RADRR, RADRL, NZ, NR, RBC
      COMMON/UPDT/LBC, ZERO, UNIT, DELTAZ, DELTAR, DELTAT, COUR, ENRGMN
      COMMON/UPDT/FLAG
C
      COMMON/EOSC/ENRG,RHO,P,T,FRAC,A,B,DEGF,RM
C
      COMMON/EDITC/NREDIT, NDELTR, ISTEP, LPRINT, TIME
C
 1000 FORMAT('1'//'  AFTER STEP NO.',I5,'  DZ=',E14.8,'  AND DT=',E14.8,
     .'  TIME=',E14.8,'  J=',I5,/)
 1001 FORMAT(2X,I3,1P9E12.4)
 1002 FORMAT(7X,'DENSITY',7X,'TEMP',6X,'PRESSURE',5X,'VELOCZ',6X,'VELOCR
     .',6X,'ENRGTOT',5X,'ZCOORD',6X,'RCOORD',5X,'HTNGRATE',/)
C
      JSTEP=ISTEP - 1
      DO 10 K=1,NREDIT
      J=1 + NDELTR*(K-1)
      WRITE(6,1000) JSTEP,DELTAZ,DELTAT,TIME,J
      WRITE(6,1002)
      WRITE(6,1001) (I,RNEW(I,J),TOLD(I,J),PNEW(I,J),VZNEW(I,J),
        VRNEW(I,J),ENEW(I,J),RADZ(I),RADR(J),FLUX(I,J), I=1,LPRINT)
   10 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE EOS
C
C     IDEAL GAS EQUATION OF STATE
C
C     OUTPUT VARIABLES
C     T IS TEMP IN KELVINS
C     P IS PRESSURE IN DYNE/CM**2
C     FRAC IS FRACTION IONIZATION, N-/NZERO
C     C IS SOUND SPEED(CM/SEC)
C     DPDE IS ENRG DERIV OF PRES AT CONST VOL
C     DPDV IS VOL DERIV OF PRES AT CONST ENRG
C
C     INPUT
C     E IS INTERNAL ENERGY, ERG/GM
C     V IS SPECIFIC VOLUME IN CC/GM
C     A = (G-*G+/GA)*(MOLWT/AVAGNO)*(2*PI*ELECMASS*BOLTZ/PLANCK**2)**1.5
C     B = IONIZATION POT/BOLTZ
C     F IS DEGREES OF FREEDOM (=3 FOR MONATOMIC GAS)
C     RM IS GAS CONSTANT IN ERG/GM/DEG
C
      COMMON/EOSC/ E, RHO, P, T, FRAC, A, B, F, RM, C
      V=1./RHO
      IF(E.GT.0.) GO TO 40
      E=2.
      WRITE(6,50)
   50 FORMAT(1X,'NEG ENERGY INPUT TO EOSIG')
      INEGE=INEGE+1
   40 CONTINUE
C
      AV=A*V
      E2OVRM=2.*E/RM
      B2=2.*B
C
      THI=E2OVRM/F
      TLO=AMAX1(0.,(E2OVRM-B2)/(F+3.))
      IF(TLO.GT.0.) GO TO 100
      FUNC1=-E2OVRM
      GO TO 110
  100 XX=AMAX1(-80.,-B/TLO)
      Y=AV*TLO*SQRT(TLO)*EXP(XX)
      X=0.5*(SQRT(Y*(Y+4.))-Y)
      FUNC1=TLO*(F+3.*X) + B2*X - E2OVRM
  110 XX=AMAX1(-80.,-B/THI)
      Y=AV*THI*SQRT(THI)*EXP(XX)
      X=0.5*(SQRT(Y*(Y+4.))-Y)
      FUNC2=THI*(F+3.*X) + B2*X - E2OVRM
      TOLD=TLO - FUNC1*(THI-TLO)/(FUNC2-FUNC1)
      TNEW=TOLD
C
      DO 10 K=1,30
      ICOUNT=K
      XX=AMAX1(-80.,-B/TOLD)
```

```
                      Y=AV*TOLD*SQRT(TOLD)*EXP(XX)
                      X=0.5*(SQRT(Y*(Y+4.))-Y)
                      FUNC=TOLD*(F + 3.*X) + B2*X - E2OVRM
                      IF(FUNC)4,20,5
                    5 THI=AMIN1(TOLD,THI)
                      GO TO 6
                    4 TLO=AMAX1(TOLD,TLO)
                    6 DXDT=(1.5 + B/TOLD)*X*(1.-X)/((2.-X)*TOLD)
                      FUNCP=F + 3.*X + (3.*TOLD + B2)*DXDT
                      TNEW=TOLD - FUNC/FUNCP
                      IF(ABS((TNEW-TOLD)/TOLD).LT.1.E-07) GO TO 20
                      IF(TNEW.GE.THI .OR. TNEW.LE.TLO) TNEW=0.5*(THI+TLO)
                      TOLD=TNEW
                   10 CONTINUE
                      WRITE(6,30) E,V
                   30 FORMAT(1X,'EOS FAILED AT E=',E14.8,2X,'V=',E14.8)
                C
                   20 CONTINUE
                      T=TNEW
                      XX=AMAX1(-80.,-B/T)
                      Y=AV*T*SQRT(T)*EXP(XX)
                      FRAC=0.5*(SQRT(Y*(Y+4.))-Y)
                      X=FRAC
                      DXDV=X*(1.-X)/(V*(2.-X))
                C
                C     CALCULATE THE PRESSURE
                C
                      P=RM*(1.+FRAC)*T/V
                C
                C     PRESSURE DERIVATIVES
                C
                      DPDT=RM*(T*DXDT + 1. + X)/V
                      DPDV=RM*T*(DXDV-(1.+X)/V)/V
                      DEDT=RM*((F+3.*X)*.5+(1.5*T+B)*DXDT)
                      DEDV=RM*DXDV*(1.5*T+B)
                C
                      DPDE=DPDT/DEDT
                      DPDV=DPDV - DPDT*DEDV/DEDT
                C
                C
                      CSQ=(V**2)*(P*DPDE-DPDV)
                      C=SQRT(CSQ)
                      RETURN
                      END
```

****** FLX ******

```
      SUBROUTINE FLX
C
C     TEMPORARY SUBROUTINE FOR 2-D CHECKOUT
C     CALCULATES HEATING RATE AT CELL CENTERS IN ERG/CM**3
C
      REAL RNEW(202,12),VZNEW(202,12),VRNEW(202,12),ENEW(202,12)
      REAL PNEW(202,12),F(202,12),TOLD(202,12),SNDSPD(202,12)
      REAL FLUX(202,12),RADZ(202),RADR(12),RBC(2,4),LBC(2,4)
      REAL ZERO(202),UNIT(202)
C
      COMMON/UPDT/ RNEW, VZNEW, VRNEW, ENEW, PNEW, F, TOLD, SNDSPD, FLUX
      COMMON/UPDT/ RADZ, RADR, RADZR, RADZL, RADRR, RADRL, NZ, NR, RBC
      COMMON/UPDT/LBC, ZERO, UNIT, DELTAZ, DELTAR, DELTAT, COUR, ENRGMN
      COMMON/UPDT/FLAG
      COMMON/DELZ/DELZL(202)
C
      COMMON/FLXC/ FZERO, XLAMDA, Z, GMWT, CUTOFF
C
      NRR=NR+1
      DO 20 J=2,NRR
      FIN=FZERO
      FLUX(NZ,J)=FIN
      JM=J-1
      NN=NZ-1
      DO 10 K=1,NN
      L=NZ+1-K
      TX=TOLD(L,JM)
      VX=1./RNEW(L,JM)
      FRAC=F(L,JM)
      XKAPA=ABSORB(TX,VX,XLAMDA,Z,FRAC,GMWT)
      IF(FIN) 2,2,3
    2 FLUX(L-1,J)=0.
      GO TO 10
    3 TERMX=AMAX1(-CUTOFF,-XKAPA*DELZL(L))
      FLUX(L-1,J)=FIN*EXP(TERMX)
      FIN=FLUX(L-1,J)
      IF((FIN/FZERO).LT.1.E-08) FIN=0.
   10 CONTINUE
      TX=TOLD(1,JM)
      VX=1./RNEW(1,JM)
      FRAC=F(1,JM)
      XKAPA=ABSORB(TX,VX,XLAMDA,Z,FRAC,GMWT)
      TERMX=AMAX1(-CUTOFF,-XKAPA*DELZL(1))
      FL=FLUX(1,J)*EXP(TERMX)
C
      DO 30 I=2,NZ
      FLUX(I,JM)=(FLUX(I,J) - FLUX(I-1,J))/DELZL(I)
   30 CONTINUE
      FLUX(1,JM)=(FLUX(1,J) - FL)/DELZL(1)
   20 CONTINUE
      RETURN
      END
```

****** FLX ******

****** TDELT ******

```
      SUBROUTINE TDELT
C
C     TWO-D, EULERIAN, CYLINDRICAL COORDINATES
C     FLAG = 1.0 FOR 1-D LAGRANGE
C
      REAL RNEW(202,12),VZNEW(202,12),VRNEW(202,12),ENEW(202,12)
      REAL PNEW(202,12),F(202,12),TOLD(202,12),SNDSPD(202,12)
      REAL FLUX(202,12),RADZ(202),RADR(12),RBC(2,4),LBC(2,4)
      REAL ZERO(202),UNIT(202)
C
      COMMON/UPDT/ RNEW, VZNEW, VRNEW, ENEW, PNEW, F, TOLD, SNDSPD, FLUX
      COMMON/UPDT/ RADZ, RADR, RADZR, RADZL, RADRR, RADRL, NZ, NR, RBC
      COMMON/UPDT/LBC, ZERO, UNIT, DELTAZ, DELTAR, DELTAT, COUR, ENRGMN
      COMMON/UPDT/FLAG
      COMMON/DELZ/DELZL(202)
C
      FACTOR=1. - FLAG
      TMIN=1.E+06
      DO 10 J=1,NR
      DO 10 I=1,NZ
      TRT=DELZL(I)/(SNDSPD(I,J) + FACTOR*ABS(VZNEW(I,J)))
      IF(TRT.GE.TMIN) GO TO 10
      TMIN=TRT
   10 CONTINUE
      DELTAT=COUR*TMIN
      RETURN
      END
```

****** UPDATE ******

```
      SUBROUTINE UPDATE
C
C     ONE TIME-CYCLE UPDATE FOR TWO-D CYLINDRICAL COORDINATES
C     (EULERIAN)    FCT TIME-SPLITTING
C     MAY BE USED FOR 1-D LAGRANGE IF FLAG IS SET TO 1.0
C
      REAL RADZOL(202)
      REAL SAVEO(202),SAVEN(202),ROLD(202),VZOLD(202),VROLD(202)
      REAL POLD(202),EOLD(202),PVZ(202),PVR(202),SCR1(202)
      REAL SCR2(202),SCR3(202),RVZOLD(202),RVZNEW(202)
      REAL RVROLD(202),RVRNEW(202)
C
      REAL RNEW(202,12),VZNEW(202,12),VRNEW(202,12),ENEW(202,12)
      REAL PNEW(202,12),F(202,12),TOLD(202,12),SNDSPD(202,12)
      REAL FLUX(202,12),RADZ(202),RADR(12),RBC(2,4),LBC(2,4)
      REAL ZERO(202),UNIT(202)
C
      COMMON/UPDT/ RNEW, VZNEW, VRNEW, ENEW, PNEW, F, TOLD, SNDSPD, FLUX
      COMMON/UPDT/ RADZ, RADR, RADZR, RADZL, RADRR, RADRL, NZ, NR, RBC
      COMMON/UPDT/LBC, ZERO, UNIT, DELTAZ, DELTAR, DELTAT, COUR, ENRGMN
      COMMON/UPDT/FLAG
      COMMON/DELZ/DELZL(202)
C
      COMMON/EOSC/ ENRG, RHO, P, T, FRAC, A, B, DEGF, RM, SND
C
      EQUIVALENCE (SAVEO(1),SAVEN(1)), (PVZ(1),PVR(1))
C
C     INITIALIZE QUANTITIES FOR CYCLE
C
      CALL NGRIDE(RADZ,NZ,RADZR,RADZL,1)
      CALL OGRIDE(NZ)
      IF(FLAG.LT.0.5) GO TO 3
      DO 4 I=1,NZ
    4 RADZOL(I)=RADZ(I)
      DTGRID=0.5*DELTAT*FLAG
      DO 2 I=1,NZ
    2 RADZ(I)=RADZOL(I) + DTGRID*VZNEW(I,1)
      RADZR=RADZR + DTGRID*VZR
      RADZL=RADZL + DTGRID*VZL
    3 CONTINUE
      CALL NGRIDE(RADZ,NZ,RADZR,RADZL,1)
C
      DO 20 J=1,NR
C
      DO 10 I=1,NZ
      ROLD(I)=RNEW(I,J)
      VZOLD(I)=VZNEW(I,J)
      VROLD(I)=VRNEW(I,J)
      RVZOLD(I)=ROLD(I)*VZOLD(I)
      RVROLD(I)=ROLD(I)*VROLD(I)
      EOLD(I)=ENEW(I,J)
      POLD(I)=PNEW(I,J)
```

```
******  UPDATE  ******
                      PVZ(I)=POLD(I)*VZOLD(I)
                   10 CONTINUE
                C
                C     HALF STEP FOR Z
                C
                      VZR=0.5*VZOLD(NZ)*(1.+RBC(1,2))
                      VZL=0.5*VZOLD(1)*(1.+LBC(1,2))
                C
                      CALL VELOCE(VZOLD,NZ,VZR,VZL,0.5*DELTAT)
                C
                      DO 21 I=1,NZ
                      SCR1(I)=-POLD(I)
                      SCR2(I)=-PVZ(I)
                      SCR3(I)=FLUX(I,J)
                   21 CONTINUE
                C
                      CALL ETBFCT(ROLD,SAVEN,NZ,RBC(1,1),LBC(1,1))
                C
                      DO 221 I=1,NZ
                  221 RNEW(I,J)=SAVEN(I)
                C
                      PLEFT=LBC(1,1)*POLD(1)
                      PRIGHT=RBC(1,1)*POLD(NZ)
                C
                      CALL SORC(NZ,0.5*DELTAT,2,UNIT,SCR1,-PRIGHT,-PLEFT)
                C
                      CALL ETBFCT(RVZOLD,RVZNEW,NZ,RBC(1,2),LBC(1,2))
                C
                      IF(NR.GT.1) GO TO 116
                      DO 130 I=1,NZ
                  130 RVRNEW(I)=0.
                      GO TO 117
                  116 CONTINUE
                C
                      CALL ETBFCT(RVROLD,RVRNEW,NZ,RBC(1,3),LBC(1,3))
                C
                  117 CONTINUE
                C
                      VLEFT=LBC(1,2)*VZOLD(1)
                      VRIGHT=RBC(1,2)*VZOLD(NZ)
                C
                      CALL SORC(NZ,0.5*DELTAT,1,ZERO,SCR2,-PRIGHT*VRIGHT,-PLEFT*VLEFT)
                C
                      CALL SORC(NZ,0.5*DELTAT,3,ZERO,SCR3,0.,0.)
                C
                      CALL ETBFCT(EOLD,SAVEN,NZ,RBC(1,4),LBC(1,4))
                C
                      DO 222 I=1,NZ
                  222 ENEW(I,J)=SAVEN(I)
                C
                C     CALCULATE INTERMEDIATE QUANTITIES
                C
                      DO 24 I=1,NZ
                      VZNEW(I,J)=RVZNEW(I)/RNEW(I,J)
```

```
******  UPDATE  ******

      VRNEW(I,J)=RVRNEW(I)/RNEW(I,J)
      ENRG=ENEW(I,J)/RNEW(I,J) - 0.5*(VZNEW(I,J)*VZNEW(I,J) +
     .   VRNEW(I,J)*VRNEW(I,J))
      IF(ENRG.LT.ENRGMN) ENRG=ENRGMN
      RHO=RNEW(I,J)
C
      CALL EOS
C
      PNEW(I,J)=P
      PVZ(I)=PNEW(I,J)*VZNEW(I,J)
   24 CONTINUE
      VZR=0.5*VZNEW(NZ,J)*(1.+RBC(1,2))
      VZL=0.5*VZNEW(1,J)*(1.+LBC(1,2))
C
      DO 223 I=1,NZ
  223 SAVEO(I)=VZNEW(I,J)
C
      IF(FLAG.LT.0.5) GO TO 5
      DTGRID=DELTAT*FLAG
      DO 6 I=1,NZ
    6 RADZ(I)=RADZOL(I) + DTGRID*VZNEW(I,1)
      RADZR=RADZR + 0.5*DTGRID*VZR
      RADZL=RADZL + 0.5*DTGRID*VZL
      CALL NGRIDE(RADZ,NZ,RADZR,RADZL,1)
    5 CONTINUE
C
      CALL VELOCE(SAVEO,NZ,VZR,VZL,DELTAT)
C
C     FULL STEP FOR Z
C
      DO 25 I=1,NZ
      SCR1(I)=-PNEW(I,J)
      SCR2(I)=-PVZ(I)
   25 CONTINUE
C
      CALL ETBFCT(ROLD,SAVEN,NZ,RBC(1,1),LBC(1,1))
C
      DO 224 I=1,NZ
  224 RNEW(I,J)=SAVEN(I)
C
      PLEFT=LBC(1,1)*PNEW(1,J)
      PRIGHT=RBC(1,1)*PNEW(NZ,J)
C
      CALL SORC(NZ,DELTAT,2,UNIT,SCR1,-PRIGHT,-PLEFT)
C
      CALL ETBFCT(RVZOLD,RVZNEW,NZ,RBC(1,2),LBC(1,2))
C
      IF(NR.EQ.1) GO TO 118
      CALL ETBFCT(RVROLD,RVRNEW,NZ,RBC(1,3),LBC(1,3))
  118 CONTINUE
C
      VLEFT=LBC(1,2)*VZNEW(1,J)
      VRIGHT=RBC(1,2)*VZNEW(NZ,J)
C
```

```
******  UPDATE  ******

      CALL SORC(NZ,DELTAT,1,ZERO,SCR2,-PRIGHT*VRIGHT,-PLEFT*VLEFT)
C
C     ADD THE TOTAL EXTERNAL SOURCE ENERGY HERE (OMIT EXTERNAL SOURCE
C     IN THE R-INTEGRATIONS)
C
      CALL SORC(NZ,DELTAT,3,ZERO,SCR3,0.,0.)
C
      CALL ETBFCT(EOLD,SAVEN,NZ,RBC(1,4),LBC(1,4))
C
      DO 225 I=1,NZ
  225 ENEW(I,J)=SAVEN(I)
C
      DO 110 I=1,NZ
      VZNEW(I,J)=RVZNEW(I)/RNEW(I,J)
  110 VRNEW(I,J)=RVRNEW(I)/RNEW(I,J)
   20 CONTINUE
C
      IF(NR.GT.1) GO TO 115
      J=1
      DO 119 I=1,NZ
      ENRG=ENEW(I,J)/RNEW(I,J) - 0.5*VZNEW(I,J)*VZNEW(I,J)
      IF(ENRG.LT.ENRGMN) ENRG=ENRGMN
      RHO=RNEW(I,J)
      CALL EOS
      PNEW(I,J)=P
      TOLD(I,J)=T
      SNDSPD(I,J)=SND
      F(I,J)=FRAC
  119 CONTINUE
C
C     EXIT FOR ONE-DIMENSIONAL CARTESIAN CALCULATION
C
      RETURN
C
  115 CONTINUE
C
C     RESET VARIABLES FOR R-INTEGRATION
C
      CALL NGRIDE(RADR,NR,RADRR,RADRL,2)
      CALL OGRIDE(NR)
      CALL NGRIDE(RADR,NR,RADRR,RADRL,2)
C
      DO 31 I=1,NZ
C
      DO 30 J=1,NR
      ROLD(J)=RNEW(I,J)
      RVZOLD(J)=RNEW(I,J)*VZNEW(I,J)
      RVROLD(J)=RNEW(I,J)*VRNEW(I,J)
      EOLD(J)=ENEW(I,J)
      VZOLD(J)=VZNEW(I,J)
      VROLD(J)=VRNEW(I,J)
      ENRG=EOLD(J)/ROLD(J) - 0.5*(VZOLD(J)*VZOLD(J) + VROLD(J)*VROLD(J))
      IF(ENRG.LT.ENRGMN) ENRG=ENRGMN
      RHO=ROLD(J)
```

```
****** UPDATE ******

C
      CALL EOS
C
      POLD(J)=P
      PVR(J)=POLD(J)*VROLD(J)
   30 CONTINUE
C
C     HALF STEP FOR R
C
      VRR=0.5*VROLD(NR)*(1.+RBC(2,2))
      VRL=0.
C
      CALL VELOCE(VROLD,NR,VRR,VRL,0.5*DELTAT)
C
      DO 33 J=1,NR
      SCR1(J)=-POLD(J)
      SCR2(J)=-PVR(J)
   33 CONTINUE
C
      CALL ETBFCT(ROLD,SAVEN,NR,RBC(2,1),LBC(2,1))
C
      DO 35 J=1,NR
   35 RNEW(I,J)=SAVEN(J)
C
      PLEFT=LBC(2,1)*POLD(1)
      PRIGHT=RBC(2,1)*POLD(NR)
C
      CALL SORC(NR,0.5*DELTAT,2,UNIT,SCR1,-PRIGHT,-PLEFT)
C
      CALL ETBFCT(RVROLD,RVRNEW,NR,RBC(2,2),LBC(2,2))
C
      CALL ETBFCT(RVZOLD,RVZNEW,NR,RBC(2,3),LBC(2,3))
C
      VLEFT=LBC(2,2)*VROLD(1)
      VRIGHT=RBC(2,2)*VROLD(NR)
C
      CALL SORC(NR,0.5*DELTAT,1,ZERO,SCR2,-PRIGHT*VRIGHT,-PLEFT*VLEFT)
C
      CALL ETBFCT(EOLD,SAVEN,NR,RBC(2,4),LBC(2,4))
C
      DO 40 J=1,NR
   40 ENEW(I,J)=SAVEN(J)
C
C     CALCULATE INTERMEDIATE QUANTITIES
C
      DO 41 J=1,NR
      VRNEW(I,J)=RVRNEW(J)/RNEW(I,J)
      VZNEW(I,J)=RVZNEW(J)/RNEW(I,J)
      ENRG=ENEW(I,J)/RNEW(I,J) - 0.5*(VZNEW(I,J)*VZNEW(I,J) + VRNEW(I,J)
     .   *VRNEW(I,J))
      IF(ENRG.LT.ENRGMN) ENRG=ENRGMN
      RHO=RNEW(I,J)
C
      CALL EOS
```

```
******  UPDATE  ******

      C
            PNEW(I,J)=P
            PVR(J)=PNEW(I,J)*VRNEW(I,J)
         41 CONTINUE
      C
            DO 42 J=1,NR
         42 SAVEO(J)=VRNEW(I,J)
            VRR=0.5*VRNEW(I,NR)*(1.+RBC(2,2))
            VRL=0.
      C
            CALL VELOCE(SAVEO,NR,VRR,VRL,DELTAT)
      C
      C     FULL STEP FOR R
      C
            DO 43 J=1,NR
            SCR1(J)=-PNEW(I,J)
            SCR2(J)=-PVR(J)
         43 CONTINUE
      C
            CALL ETBFCT(ROLD,SAVEN,NR,RBC(2,1),LBC(2,1))
      C
            DO 45 J=1,NR
         45 RNEW(I,J)=SAVEN(J)
      C
            PLEFT=LBC(2,1)*PNEW(I,1)
            PRIGHT=RBC(2,1)*PNEW(I,NR)
      C
            CALL SORC(NR,DELTAT,2,UNIT,SCR1,-PRIGHT,-PLEFT)
      C
            CALL ETBFCT(RVROLD,RVRNEW,NR,RBC(2,2),LBC(2,2))
      C
            CALL ETBFCT(RVZOLD,RVZNEW,NR,RBC(2,3),LBC(2,3))
      C
            VLEFT=LBC(2,2)*VRNEW(I,1)
            VRIGHT=RBC(2,2)*VRNEW(I,NR)
      C
            CALL SORC(NR,DELTAT,1,ZERO,SCR2,-PRIGHT*VRIGHT,-PLEFT*VLEFT)
      C
            CALL ETBFCT(EOLD,SAVEN,NR,RBC(2,4),LBC(2,4))
      C
            DO 50 J=1,NR
         50 ENEW(I,J)=SAVEN(J)
      C
            DO 70 J=1,NR
            VZNEW(I,J)=RVZNEW(J)/RNEW(I,J)
            VRNEW(I,J)=RVRNEW(J)/RNEW(I,J)
            RHO=RNEW(I,J)
            ENRG=ENEW(I,J)/RNEW(I,J) - 0.5*(VZNEW(I,J)*VZNEW(I,J) + VRNEW(I,J)
           .   *VRNEW(I,J))
            IF(ENRG.LT.ENRGMN) ENRG=ENRGMN
      C
            CALL EOS
      C
            PNEW(I,J)=P
```

```
******  UPDATE  ******

  323.                      TOLD(I,J)=T
  324.                      F(I,J)=FRAC
  325.                      SNDSPD(I,J)=SND
  326.                   70 CONTINUE
  327.                C
  328.                   31 CONTINUE
  329.                C
  330.                      RETURN
  331.                      END
```

```
***** CONSRE *****

      1.                SUBROUTINE CONSRE (RHO, N, CSUM)
      2.          C
      3.                REAL RHO(N)
      4.          C
      5.          C     COMPUTE VOLUME INTEGRAL OF DENSITY OVER GRID
      6.          C     ARGUMENTS
      7.          C     RHO      DENSITY                                        (I)
      8.          C     N        NUMBER OF INTERIOR GRID POINTS                 (I)
      9.          C     CSUM     OSTENSIBLY CONSERVED TOTAL MASS                (O)
     10.          C
     11.          C
     12.                LOGICAL LSOURC
     13.                REAL SOURCE(202), SCRH(202), RHOT(202), DIFF(202)
     14.                REAL ADUGTH(202), FLXH(202), NULH(202), MULH(202)
     15.                REAL LNRHOT(202), FSGN(202), FABS(202), EPSH(202)
     16.                REAL LORHOT(202), TERP(202), TERM(202), ADUDTH(202)
     17.                REAL LO(202), LN(202), LH(202), RLO(202), RLN(202)
     18.                REAL RNH(202), ROH(202), RLH(202), AH(202)
     19.                COMMON /COM1/ SOURCE, SCRH, RHOT, DIFF, ADUGTH, NULH, MULH
     20.                COMMON /COM1/ LORHOT, ADUDTH, LO, LN, LH, RLO, RLN, RNH, ROH
     21.                COMMON /COM1/ RLH, AH, LSOURC
     22.                EQUIVALENCE (EPSH(1), SCRH(1)), (LNRHOT(1), LORHOT(1))
     23.                EQUIVALENCE (FLXH(1), SCRH(1)), (FSGN(1), RHOT(1))
     24.                EQUIVALENCE (FABS(1), SCRH(1)), (TERP(1), SOURCE(1))
     25.                EQUIVALENCE (TERM(1), SOURCE(1))
     26.          C
     27.          C
     28.                CSUM = 0.0
     29.                DO 501 I = 1, N
     30.              501 CSUM = CSUM + LN(I) * RHO(I)
     31.                RETURN
     32.                END
```

```
      SUBROUTINE ETBFCT (RHOO, RHON, N, RBC, LBC)
C
      REAL RHOO(N), RHON(N), RBC, LBC
C
C        FLUX-CORRECTED TRANSPORT FOR NONPERIODIC B. C.
C     SOLVE CONTINUITY EQUATION
C     (D/DT) RHO = -(1/A)*(D/DR) (A*V*RHO) + (1/A)*(D/DR) (A*D1)
C        + C*(D/DR) D2 + D3      WHERE A = R**(ALPHA-1)
C
C     ARGUMENTS
C     RHOO     GRID POINT DENSITIES AT START OF STEP                (I)
C     RHON     GRID POINT DENSITIES AT END OF STEP                  (O)
C     N        NUMBER OF INTERIOR GRID POINTS                       (I)
C     RBC      RIGHT BOUNDARY FACTOR                                (I)
C     RBC      LEFT  BOUNDARY FACTOR                                (I)
C
C
      LOGICAL LSOURC
      REAL SOURCE(202), SCRH(202), RHOT(202), DIFF(202)
      REAL ADUGTH(202), FLXH(202), NULH(202), MULH(202)
      REAL LNRHOT(202), FSGN(202), FABS(202), EPSH(202)
      REAL LORHOT(202), TERP(202), TERM(202), ADUDTH(202)
      REAL LO(202), LN(202), LH(202), RLO(202), RLN(202)
      REAL RNH(202), ROH(202), RLH(202), AH(202)
      COMMON /COM1/ SOURCE, SCRH, RHOT, DIFF, ADUGTH, NULH, MULH
      COMMON /COM1/ LORHOT, ADUDTH, LO, LN, LH, RLO, RLN, RNH, ROH
      COMMON /COM1/ RLH, AH, LSOURC
      EQUIVALENCE (EPSH(1), SCRH(1)), (LNRHOT(1), LORHOT(1))
      EQUIVALENCE (FLXH(1), SCRH(1)), (FSGN(1), RHOT(1))
      EQUIVALENCE (FABS(1), SCRH(1)), (TERP(1), SOURCE(1))
      EQUIVALENCE (TERM(1), SOURCE(1))
C
C
C     CALCULATE CONVECTIVE AND DIFFUSIVE FLUXES
      NP = N + 1
      DO 11 I = 2, N
      FLXH(I) = 0.5 * ADUDTH(I) * (RHOO(I) + RHOO(I-1))
   11 DIFF(I) = NULH(I) * (RHOO(I) - RHOO(I-1))
      RHOL = RHOO(1) * LBC
      RHOR = RHOO(N) * RBC
      DIFF(1) = NULH(1) * (RHOO(1) - RHOL)
      DIFF(NP) = NULH(NP) * (RHOR - RHOO(N))
      FLXH(1) = 0.5 * ADUDTH(1) * (RHOO(1) + RHOL)
      FLXH(NP) = 0.5 * ADUDTH(NP) * (RHOR + RHOO(N))
C
C     CALCULATE LAMBDAO*RHOT, THE TRANSPORTED MASS ELEMENTS
      DO 12 I = 1, N
   12 LORHOT(I) = LO(I) * RHOO(I) - FLXH(I+1) + FLXH(I)
C
C     ADD SOURCE TERMS
      IF (.NOT. LSOURC) GO TO 14
      DO 13 I = 1, N
```

```
******  ETBFCT  ******

                  13 LORHOT(I) = LORHOT(I) + SOURCE(I)
                C
                C     CALCULATE ANTIDIFFUSIVE FLUXES
                  14 DO 16 I = 1, N
                  16 RHOT(I) = LORHOT(I) * RLO(I)
                     DO 17 I = 2, N
                  17 FLXH(I) = MULH(I) * (RHOT(I) - RHOT(I-1))
                     FLXH(1) = MULH(1) * (RHOT(1) - LBC * RHOT(1))
                     FLXH(NP) = MULH(NP) * (RBC * RHOT(N) - RHOT(N))
                C
                C     CALCULATE DIFFUSED MASS ELEMENTS
                     DO 18 I = 1, N
                  18 LNRHOT(I) = LORHOT(I) + DIFF(I+1) - DIFF(I)
                C
                C     CALCULATE TRANSPORTED/DIFFUSED DENSITY AND FLUX
                     DO 19 I = 1, N
                  19 RHOT(I) = LNRHOT(I) * RLN(I)
                     DO 20 I = 2, N
                  20 DIFF(I) = RHOT(I) - RHOT(I-1)
                     DIFF(1) = RHOT(1) - LBC * RHOT(1)
                     DIFF(NP) = RBC * RHOT(N) - RHOT(N)
                C
                C     CALCULATE SIGN OF DIFFUSED/TRANSPORTED FLUX
                     DO 21 I = 1, NP
                  21 FSGN(I) = SIGN (1.0, DIFF(I))
                C
                C     CALCULATE MAGNITUDE OF ANTIDIFFUSIVE FLUX
                     DO 22 I = 1, NP
                  22 FABS(I) = ABS(FLXH(I))
                C
                C     CALCULATE FLUX LIMITS ON RIGHT AND LEFT
                     DO 23 I = 1, N
                  23 TERP(I) = FSGN(I) * LN(I) * DIFF(I+1)
                     TERP(NP) = 1.0E+30
                     DO 24 I = 1, NP
                  24 FABS(I) = AMIN1 (TERP(I), FABS(I))
                     DO 25 I = 2, NP
                  25 TERM(I) = FSGN(I) * LN(I-1) * DIFF(I-1)
                     TERM(1) = 1.0E+30
                C    CORRECT FLUXES
                     DO 35 I = 1, NP
                  35 DIFF(I) = AMIN1 (FABS(I), TERM(I))
                     DO 36 I = 1, NP
                  36 FLXH(I) = AMAX1 (0.0, DIFF(I))
                     DO 37 I = 1, NP
                  37 FLXH(I) = FSGN(I) * FLXH(I)
                C
                C     CALCULATE FLUX-CORRECTED DENSITIES
                     DO 40 I = 1, N
                  40 LNRHOT(I) = LNRHOT(I) - FLXH(I+1) + FLXH(I)
                     DO 41 I = 1, N
                     SOURCE(I) = 0.0
                  41 RHON(I) = LNRHOT(I) * RLN(I)
                     LSOURC = .FALSE.
```

```
******  ETBFCT  ******

  107.                    RETURN
  108.                    END
```

```
                      SUBROUTINE NGRIDE (RADN, N, RADR, RADL, ALPHA)
                C
                      INTEGER ALPHA
                      REAL RADN(N)
                C
                C     SETUP FOR NEW GEOMETRY VARIABLES
                C     ARGUMENTS
                C     RADN     NEW GRID-POINT POSITIONS                          (I)
                C     N        NUMBER OF INTERIOR GRID POINTS                    (I)
                C     RADR     POSITION OF RIGHT BOUNDARY                        (I)
                C     RADL     POSITION OF  LEFT BOUNDARY                        (I)
                C     ALPHA    = 1  FOR CARTESIAN GEOMETRY                       (I)
                C              = 2  FOR CYLINDRICAL GEOMETRY                     (I)
                C              = 3  FOR SPHERICAL GEOMETRY                       (I)
                C
                C
                      LOGICAL LSOURC
                      REAL SOURCE(202), SCRH(202), RHOT(202), DIFF(202)
                      REAL ADUGTH(202), FLXH(202), NULH(202), MULH(202)
                      REAL LNRHOT(202), FSGN(202), FABS(202), EPSH(202)
                      REAL LORHOT(202), TERP(202), TERM(202), ADUDTH(202)
                      REAL LO(202), LN(202), LH(202), RLO(202), RLN(202)
                      REAL RNH(202), ROH(202), RLH(202), AH(202)
                      COMMON /COM1/ SOURCE, SCRH, RHOT, DIFF, ADUGTH, NULH, MULH
                      COMMON /COM1/ LORHOT, ADUDTH, LO, LN, LH, RLO, RLN, RNH, ROH
                      COMMON /COM1/ RLH, AH, LSOURC
                      EQUIVALENCE (EPSH(1), SCRH(1)), (LNRHOT(1), LORHOT(1))
                      EQUIVALENCE (FLXH(1), SCRH(1)), (FSGN(1), RHOT(1))
                      EQUIVALENCE (FABS(1), SCRH(1)), (TERP(1), SOURCE(1))
                      EQUIVALENCE (TERM(1), SOURCE(1))
                C
                C
                      COMMON/DELZ/DELZL(202)
                      DATA PI, FTPI /3.1415927, 4.1887902/
                C
                      NP = N + 1
                C     CALCULATE NEW INTERFACE POSITIONS
                      DO 202 I = 2, N
                  202 RNH(I) = 0.5 * (RADN(I) + RADN(I-1))
                      RNH(1) = RADL
                      RNH(NP) = RADR
                C
                C     SELECT COORDINATE SYSTEM
                      GO TO (203, 206, 209), ALPHA
                C
                C     CARTESIAN COORDINATES
                  203 DO 204 I = 1, NP
                  204 AH(I) = 1.0
                      DO 205 I = 1, N
                      LN(I)=RNH(I+1) - RNH(I)
                  205 DELZL(I)=LN(I)
                      GO TO 213
```

```
******  NGRIDE  ******

      C
      C     CYLINDRICAL COORDINATES
  206   DO 207 I = 1, NP
        DIFF(I) = RNH(I) * RNH(I)
  207   AH(I) = PI * (ROH(I) + RNH(I))
        DO 208 I = 1, N
  208   LN(I) = PI * (DIFF(I+1) - DIFF(I))
        GO TO 213
      C
      C     SPHERICAL COORDINATES
  209   DO 210 I = 1, NP
        DIFF(I) = RNH(I) * RNH(I) * RNH(I)
  210   SCRH(I) = (ROH(I) + RNH(I)) * ROH(I)
        DO 211 I = 1, NP
  211   AH(I) = FTPI * (SCRH(I) + RNH(I) * RNH(I))
        DO 212 I = 1, N
  212   LN(I) = FTPI * (DIFF(I+1) - DIFF(I))
      C
      C     REMAINING GEOMETRIC VARIABLES
  213   DO 214 I = 2, N
  214   LH(I) = 0.5 * (LN(I) + LN(I-1))
        LH(1) = LN(1)
        LH(NP) = LN(N)
        DO 215 I = 1, N
  215   RLN(I) = 1.0 / LN(I)
        DO 216 I = 1, NP
  216   ADUGTH(I) = AH(I) * (RNH(I) - ROH(I))
        DO 217 I = 2, N
  217   RLH(I) = 0.5 * (RLN(I) + RLN(I-1))
        RLH(1) = RLN(1)
        RLH(NP) = RLN(N)
        RETURN
        END
```

```
      SUBROUTINE OGRIDE (N)
C
C     SETUP FOR OLD GEOMETRY VARIABLES
C     ARGUMENTS
C     N       NUMBER OF INTERIOR GRID POINTS                    (I)
C
C
      LOGICAL LSOURC
      REAL SOURCE(202), SCRH(202), RHOT(202), DIFF(202)
      REAL ADUGTH(202), FLXH(202), NULH(202), MULH(202)
      REAL LNRHOT(202), FSGN(202), FABS(202), EPSH(202)
      REAL LORHOT(202), TERP(202), TERM(202), ADUDTH(202)
      REAL LO(202), LN(202), LH(202), RLO(202), RLN(202)
      REAL RNH(202), ROH(202), RLH(202), AH(202)
      COMMON /COM1/ SOURCE, SCRH, RHOT, DIFF, ADUGTH, NULH, MULH
      COMMON /COM1/ LORHOT, ADUDTH, LO, LN, LH, RLO, RLN, RNH, ROH
      COMMON /COM1/ RLH, AH, LSOURC
      EQUIVALENCE (EPSH(1), SCRH(1)), (LNRHOT(1), LORHOT(1))
      EQUIVALENCE (FLXH(1), SCRH(1)), (FSGN(1), RHOT(1))
      EQUIVALENCE (FABS(1), SCRH(1)), (TERP(1), SOURCE(1))
      EQUIVALENCE (TERM(1), SOURCE(1))
C
C
      DATA ROH /202*1.0/
C
      NP = N + 1
      DO 401 I = 1, N
      LO(I) = LN(I)
  401 RLO(I) = RLN(I)
      DO 402 I = 1, NP
  402 ROH(I) = RNH(I)
      RETURN
      END
```

****** SORC ******

```
      1.                SUBROUTINE SORC (N, DT, MODES, C, D, DR, DL)
      2.          C
      3.                REAL C(N), D(N)
      4.          C
      5.          C     CALCULATE SOURCE CONTRIBUTIONS
      6.          C     ARGUMENTS
      7.          C     N        NUMBER OF INTERIOR GRID POINTS                    (I)
      8.          C     DT       STEPSIZE FOR TIME INTEGRATION                     (I)
      9.          C     MODES    =1  ADDS  + DIV(D)                                (I)
     10.          C              =2  ADDS  + C * GRAD(D)                           (I)
     11.          C              =3  ADDS  + D                                     (I)
     12.          C     C        SOURCE VARIABLE AT GRID POINTS                    (I)
     13.          C     D        SOURCE VARIABLE AT GRID POINTS                    (I)
     14.          C     DR       D AT RIGHT BOUNDARY                               (I)
     15.          C     DL       D AT  LEFT BOUNDARY                               (I)
     16.          C
     17.          C
     18.                LOGICAL LSOURC
     19.                REAL SOURCE(202), SCRH(202), RHOT(202), DIFF(202)
     20.                REAL ADUGTH(202), FLXH(202), NULH(202), MULH(202)
     21.                REAL LNRHOT(202), FSGN(202), FABS(202), EPSH(202)
     22.                REAL LORHOT(202), TERP(202), TERM(202), ADUDTH(202)
     23.                REAL LO(202), LN(202), LH(202), RLO(202), RLN(202)
     24.                REAL RNH(202), ROH(202), RLH(202), AH(202)
     25.                COMMON /COM1/ SOURCE, SCRH, RHOT, DIFF, ADUGTH, NULH, MULH
     26.                COMMON /COM1/ LORHOT, ADUDTH, LO, LN, LH, RLO, RLN, RNH, ROH
     27.                COMMON /COM1/ RLH, AH, LSOURC
     28.                EQUIVALENCE (EPSH(1), SCRH(1)), (LNRHOT(1), LORHOT(1))
     29.                EQUIVALENCE (FLXH(1), SCRH(1)), (FSGN(1), RHOT(1))
     30.                EQUIVALENCE (FABS(1), SCRH(1)), (TERP(1), SOURCE(1))
     31.                EQUIVALENCE (TERM(1), SOURCE(1))
     32.          C
     33.          C
     34.                DATA LSOURC, SOURCE /.FALSE., 202*0.0/
     35.          C
     36.                NP = N + 1
     37.                DTH = 0.5 * DT
     38.                DTQ = 0.25 * DT
     39.                GO TO (310,320,330,340) MODES
     40.          C
     41.          C     ADD DIV(D)
     42.            310 DO 311 I = 2, N
     43.            311 SCRH(I) = DTH * AH(I) * (D(I) + D(I-1))
     44.                SCRH(1) = DT * AH(1) * DL
     45.                SCRH(NP) = DT * AH(NP) * DR
     46.                DO 312 I = 1, N
     47.            312 SOURCE(I) = SOURCE(I) + SCRH(I+1) - SCRH(I)
     48.                LSOURC = .TRUE.
     49.                RETURN
     50.          C
     51.          C     ADD C * GRAD(D)
     52.            320 DO 321 I = 2, N
```

```
******  SORC  ******

                  321 SCRH(I) = DTQ * (D(I) + D(I-1))
                      SCRH(1) = DTH * DL
                      SCRH(NP) = DTH * DR
                      DO 322 I = 1, N
                  322 DIFF(I) = SCRH(I+1) - SCRH(I)
                      DO 323 I = 1, N
                  323 SOURCE(I) = SOURCE(I) + C(I) * (AH(I+1) + AH(I)) * DIFF(I)
                      LSOURC = .TRUE.
                      RETURN
                C
                C     ADD D
                  330 DO 331 I = 1, N
                  331 SOURCE(I) = SOURCE(I) + DT * LQ(I) * D(I)
                      LSOURC = .TRUE.
                      RETURN
                C
                C     FLUX SOURCE USING VALUES AT ZONE INTERFACES
                C
                  340 DO 341 I=2,N
                      SOURCE(I)=(D(I) - D(I-1))*DT
                  341 CONTINUE
                      SOURCE(1)=(D(1) - DL)*DT
                      RETURN
                      END
```

```
      1                 SUBROUTINE VELOCE (U, N, UR, UL, DT)
      2               C
      3                 REAL U(N)
      4               C VELOCITY DEPENDENT VARIABLE CALCULATION
      5               C ARGUMENTS
      6               C U       FLOW VELOCITY AT GRID POINTS                    (I)
      7               C N       NUMBER OF INTERIOR GRID POINTS                  (I)
      8               C UR      VELOCITY AT RIGHT BOUNDARY                      (I)
      9               C UL      VELOCITY AT  LEFT BOUNDARY                      (I)
     10               C DT      STEPSIZE FOR TIME INTEGRATION                   (I)
     11               C
     12               C
     13                 LOGICAL LSOURC
     14                 REAL SOURCE(202), SCRH(202), RHOT(202), DIFF(202)
     15                 REAL ADUGTH(202), FLXH(202), NULH(202), MULH(202)
     16                 REAL LNRHOT(202), FSGN(202), FABS(202), EPSH(202)
     17                 REAL LORHOT(202), TERP(202), TERM(202), ADUDTH(202)
     18                 REAL LO(202), LN(202), LH(202), RLO(202), RLN(202)
     19                 REAL RNH(202), ROH(202), RLH(202), AH(202)
     20                 COMMON /COM1/ SOURCE, SCRH, RHOT, DIFF, ADUGTH, NULH, MULH
     21                 COMMON /COM1/ LORHOT, ADUDTH, LO, LN, LH, RLO, RLN, RNH, ROH
     22                 COMMON /COM1/ RLH, AH, LSOURC
     23                 EQUIVALENCE (EPSH(1), SCRH(1)), (LNRHOT(1), LORHOT(1))
     24                 EQUIVALENCE (FLXH(1), SCRH(1)), (FSGN(1), RHOT(1))
     25                 EQUIVALENCE (FABS(1), SCRH(1)), (TERP(1), SOURCE(1))
     26                 EQUIVALENCE (TERM(1), SOURCE(1))
     27               C
     28               C
     29                 NP = N + 1
     30                 DTH = 0.5 * DT
     31               C
     32               C     CALCULATE INTERFACE AREA * VELOCITY DIFFERENCE * DT
     33                 DO 101 I = 2, N
     34               101 ADUDTH(I) = AH(I) * DTH * (U(I) + U(I-1)) - ADUGTH(I)
     35                 ADUDTH(1) = AH(1) * DT * UL - ADUGTH(1)
     36                 ADUDTH(NP) = AH(NP) * DT * UR - ADUGTH(NP)
     37               C
     38               C     CALCULATE HALF-CELL EPSILON = V*DT/DX
     39                 DO 102 I = 1, NP
     40               102 EPSH(I) = ADUDTH(I) * RLH(I)
     41               C
     42               C     CALCULATE DIFFUSION AND ANTIDIFFUSION COEFFICIENTS
     43                 DO 103 I = 1, NP
     44                 NULH(I) = 0.166666667 + 0.333333333 * EPSH(I) * EPSH(I)
     45               103 MULH(I) = 0.25 - 0.5 * NULH(I)
     46                 DO 104 I = 1, NP
     47                 NULH(I) = LH(I) * NULH(I)
     48               104 MULH(I) = LH(I) * MULH(I)
     49                 RETURN
     50                 END
```

****** VELOCE ******

APPENDIX II

LISTING OF LASER BEAM PROPAGATION ROUTINES

```
****** ALPHA ******

      1.                    SUBROUTINE ALPHA
      2.                    COMMON A(30,50)
      3.                    COMMON /N1/ N,MIND,IH,IA,IB,IC,ID,IE,IF,IZ
      4.                    COMMON /N2/ IW,IX,IGP,IGM,IGBB,IGB,IG,IRBB,IRB,IR,IRP,IMU,IAB,
      5.                   . IALPHA,IYP,II
      6.                C
      7.                C     GIVEN ABSORPTION COEFFICIENT AB  AT TRAJECTORY POINTS R(J)
      8.                C     COMPUTE ALPHA=INTENSITY-WEIGHTED MEAN BETWEEN R(J) AND R(J+1)
      9.                C
     10.                    DO 5 J=1,N
     11.                  5 A(IA,J)=A(IAB,J)
     12.                    CALL SPLN5B
     13.                    DO 30 J=1,N
     14.                    H=A(IH,J)
     15.                    IF(H.LT.1.E-3) GO TO 10
     16.                    EXPMH=EXP(-H)
     17.                    D=1.-EXPMH
     18.                    GO TO 20
     19.                 10 D=H*(1.-H/2.*(1.-H/3.*(1.-H/4.*(1.-H/5.))))
     20.                    EXPMH=1.-D
     21.                 20 SUMA=A(IF,J)
     22.                    SUMB=SUMA*H**5
     23.                    SUMA=5.*SUMA+A(IE,J)
     24.                    SUMB=SUMB+SUMA*H**4
     25.                    SUMA=4.*SUMA+A(ID,J)
     26.                    SUMB=SUMB+SUMA*H**3
     27.                    SUMA=3.*SUMA+A(IC,J)
     28.                    SUMB=SUMB+SUMA*H*H
     29.                    SUMA=2.*SUMA+A(IB,J)
     30.                    SUMB=SUMB+SUMA*H
     31.                    SUMA=SUMA+A(IA,J)
     32.                    A(IALPHA,J)=SUMA-SUMB*EXPMH/D
     33.                 30 CONTINUE
     34.                    RETURN
     35.                    END
```

****** DIFS ******

```
      SUBROUTINE DIFS (HB,HF,YPPB,YPP,YPPF,YB,Y,YF,ITER)
C     EVALUATES PREDICTOR AND CORRECTOR FORMULAS FOR INTEGRATION OF
C           (D/DZ)**2  Y  =  G(Z,Y)
C     WITH YPP=G(Z,Y),YPPB=G(Z-HB,YB), ETC.   OUTPUT IS YF=Y(Z+HF).
      HBF=HB+HF
      HBHF=HB*HF
      HB2=HB*HB
      HF2=HF*HF
      HBYF = HBF*Y - HF*YB
      IF (ITER.GT.0) GO TO 10
C     PREDICTOR
      T1=0.5*HBHF*HBF
      T2=HF/6.*(HF2-HB2)
      HBYF=HBYF + T1*YPP + T2*(YPP-YPPB)
      GO TO 20
C     CORRECTOR
   10 T3 = HF*(HB*HBF - HF2)
      T4 = HBF*(HBF*HBF + HBHF)
      T5 = HB*(HF*HBF - HB2)
      HBYF = HBYF + (T3*YPPB + T4*YPP + T5*YPPF)/12.
   20 YF = HBYF/HB
      RETURN
      END
```

```
***** EDIS *****

      1.                    SUBROUTINE EDIS (RHO,AWT,TEK,ZBAR,ZINDX,WL,EABS,EMU)
      2.                    COMMON /C/ Z,HB,HF,T,AKZ,AKZ2
      3.              C     FOR MASS DENSITY RHO (GRAMS/CM**3), ATOMIC WEIGHT AWT (AMU),
      4.              C     ELECTRON TEMPERATURE TEK (KELVINS), IONIZATION ZBAR (FREE
      5.              C     ELECTRONS PER ATOM), UNPERTURBED REFRACTIVE INDEX ZINDX,
      6.              C     RADIATION WAVELENGTH WL (CM), COMPUTE ABSORPTION COEFFICIENT
      7.              C     EABS (CM**-1) AND INDEX PERTURBATION EMU DUE TO FREE ELECTRONS.
      8.              C
      9.                    COMPLEX KSQ, KSTAR
     10.              C
     11.                    DATA TWOPI /6.283185/
     12.                    DATA C /2.9979E10/
     13.              C     OPCON = 4*PI*E**2/M  (SEC**-2 * CM**3,  FOR ELECTRONS)
     14.                    DATA OPCON /3.1826E9/
     15.              C     VEV = MEAN VELOCITY OF 1 EV ELECTRONS
     16.                    DATA VEV /6.692349E07/
     17.              C     RVEV = MEAN RECIPROCAL VELOCITY OF 1 EV ELECTRONS
     18.                    DATA RVEV /1.9025296E-08/
     19.              C     ATOMIC MASS UNIT (GRAMS)
     20.                    DATA AWTU /1.66053E-24/
     21.              C     KELVINS PER EV
     22.                    DATA EVK /11604.5/
     23.              C     WAVELENGTH FOR 1 EV
     24.                    DATA WLEV /1.23985E-04/
     25.              C     ELECTRON CHARGE SQUARED, CM*EV
     26.                    DATA ESQ /1.439978E-07/
     27.              C     PI/SQRT(27)
     28.                    DATA PIOR27 /.6045997/
     29.              C
     30.              C     PHOTON ENERGY, EV
     31.                    HNU=WLEV/WL
     32.              C     VACUUM WAVE NUMBER
     33.                    WOC = TWOPI/WL
     34.              C     CIRCULAR FREQUENCY
     35.                    W = WOC * C
     36.              C     UNPERTURBED WAVE NUMBER
     37.                    AKZ = ZINDX * WOC
     38.                    AKZ2 = AKZ*AKZ
     39.              C     ATOM DENSITY
     40.                    ANA = RHO/(AWTU*AWT)
     41.              C     ELECTRON DENSITY
     42.                    ANE = ZBAR * ANA
     43.              C     ION DENSITY
     44.                    ANI = AMIN1 (ANA,ANE)
     45.              C     Z SQUARED AVERAGE, APPROX.
     46.                    ZSQ = AMAX1 (1.,ZBAR**2)
     47.              C     NEUTRAL DENSITY
     48.                    ANN = ANA - ANI
     49.              C     ELECTRON TEMPERATURE, VELOCITY, RECIPROCAL VELOCITY
     50.                    TEV = TEK / EVK
     51.                    SQRTT = SQRT(TEV)
     52.                    VE = VEV * SQRTT

65
```

```
***** EDIS *****

  53.                   RVE = RVEV / SQRTT
  54.             C     ELECTRON TRANSPORT CROSS SECTION OF NEUTRALS
  55.             C        ASSUME 10 * PI*A0**2
  56.                   SIGM = 8.797E35E-16
  57.             C     ELECTRON-NEUTRAL COLLISION FREQUENCY
  58.                   ANUN = ANN * VE * SIGM
  59.             C     PLASMA FREQUENCY SQUARED
  60.                   OP2 = OPCON * ANE
  61.             C     ION COLLISION FREQUENCY FOR TRANSFER OF HNU
  62.                   ANUI = PIOR27*RVE*ZSQ*ESQ/HNU*OPCON*ANI*(1.-EXP(-HNU/TEV))
  63.             C     COMPLEX WAVE NUMBER, EABS, EMU
  64.                   KSQ = AKZ2 - OP2/C**2 * W/CMPLX(W,ANUN+ANUI)
  65.                   KSTAR = CSQRT(KSQ)
  66.                   EABS = 2.*AIMAG(KSTAR)
  67.                   EMU = REAL(KSTAR)/AKZ - 1.
  68.             C
  69.                   RETURN
  70                    END
```

****** GRADM ******

```
      SUBROUTINE GRADM
      COMMON A(30,50)
      COMMON /N1/ N,MIND,IH,IA,IB,IC,ID,IE,IF,IZ
      COMMON /N2/ IW,IX,IGP,IGM,IGBB,IGB,IG,IRBB,IRB,IR,IRP,IMU,IAB,
     . IALPHA,IYP,II
      DO 10 J=1,N
   10 A(IA,J)=A(IMU,J)
      CALL SPLN5B
      DO 20 J=1,N
   20 A(IGM,J)=A(IB,J)*A(IR,J)/A(IYP,J)
      RETURN
      END
```

****** GRADP ******

```
      SUBROUTINE GRADP
      COMMON A(30,50)
      COMMON /N1/ N,MIND,IH,IA,IB,IC,ID,IE,IF,IZ
      COMMON /N2/ IW,IX,IGP,IGM,IGBB,IGB,IG,IRBB,IRB,IR,IRP,IMU,IAB,
     . IALPHA,IYP,II
      COMMON /C/ Z,HB,HF,T,AKZ,AKZ2
C
C     GIVEN TRAJECTORY RADIUS R(J), J=1,N, CALCULATE Y(J)=R(J)**2/2
C     AND ITS DERIVATIVES BY X, THEN EVALUATE GRADP, THE DERIVATIVE BY
C     R OF THE DIFFRACTION POTENTIAL
C           P = 1/(4*K**2)*((LAPLACIAN I)/I - .5*((GRAD I)/I)**2)
C
      DO 5 J=1,N
    5 A(IA,J)=.5*A(IR,J)**2
      CALL SPLN5B
      DO 10 J=1,N
      Y=A(IA,J)
      Y1P=A(IB,J)
      Y2P=A(IC,J)*2.
      Y3P=A(ID,J)*6.
      Y4P=A(IE,J)*24.
C     T1=(DI/DY)/I
C     T2=DT1/DY
C     T3=DT2/DY
      T1=-(1.+Y2P/Y1P)/Y1P
      T2=(Y2P-Y3P+2.*Y2P**2/Y1P)/Y1P**3
      T3=(Y3P-Y4P)/Y1P**3+(7.*Y2P*Y3P-4.*Y2P**2)/Y1P**5-8.*Y2P**3/Y1P**6
C     A(IP,J)=(2.*T1 + Y*T1**2 + 2.*Y*T2)/(4.*AKZ2)
      A(IGP,J)=A(IR,J)/(4.*AKZ2)*(4.*T2+T1**2+2.*Y*(T1*T2+T3))
      A(IYP,J)=Y1P
   10 CONTINUE
      RETURN
      END
```

```
 ****** SPLN5A ******

ELT S7 RS70-1 02/02-16:29:42-(0,)
     1.                      SUBROUTINE SPLN5A
     2.                      COMMON A(30,50)
     3.                      DATA NRDIM/30/
     4.                      COMMON /N1/ N,MIND,IH,IA,IB,IC,ID,IE,IF,IZ
     5.                C
     6.                C     SETUP PROCEDURE FOR PENTADIAGONAL SYMMETRIC COEFFICIENT MATRIX
     7.                C
     8.                C
     9.                      MAXD = MIND+2
    10.                      DO 5 I=MIND,MAXD
    11.                      DO 2 J=1,N
    12.                    2 A(I,J) = 0.
    13.                    5 CONTINUE
    14.                      M=N-2
    15.                      Q = A(IH,1)
    16.                      R = A(IH,2)
    17.                      Q2 = Q*Q
    18.                      Q3 = Q2*Q
    19.                      R2 = R*R
    20.                      R3 = R2*R
    21.                      QR = Q+R
    22.                      A(MIND,1)=1.
69  23.                      DO 10 I=2,M
    24.                      P=Q
    25.                      Q=R
    26.                      R = A(IH,I+1)
    27.                      P2=Q2
    28.                      Q2=R2
    29.                      R2 = R*R
    30.                      P3=Q3
    31.                      Q3=R3
    32.                      R3 = R2*R
    33.                      PQ=QR
    34.                      QR = Q+R
    35.                      PR=P*R
    36.                      PQQR = PQ*QR
    37.                      A(MIND,I) = 6.*P3/PQ**2  + Q*(30.*PR*PR+(P+R)*Q*(40.*PR+14.*Q2) +
    38.                     .Q2*(16.*(P2+R2)+42.*PR+4.*Q2))/PQQR**2 + 6.*R3/QR**2
    39.                      A(MIND+1,I) = Q2*(P*QR + 3.*PQ*(QR+R+R))/(PQQR*QR)
    40.                      A(MIND+1,I-1)=A(MIND+1,I-1)+Q2*(R*PQ + 3.*QR*(PQ+P+P))/(PQQR*PQ)
    41.                      A(MIND+2,I-1) = Q3/PQQR
    42.                   10 CONTINUE
    43.                      A(MIND+1,1)=0.
    44.                      A(MIND+1,M)=0.
    45.                      A(MIND+2,1)=0.
    46.                      A(MIND+2,M-1) = 0.
    47.                      MAXD=MIND + 2
    48.                      CALL SYMFAC (NRDIM,M,MIND,MAXD,A,JR)
    49.                      IF (JR.EQ.0) RETURN
    50.                C     CALL ERROR
    51.                      RETURN
    52.                      END
```

****** SPLNSA ******

```
      SUBROUTINE SPLN5B
      COMMON A(30,50)
      DATA NRDIM/30/
      COMMON /N1/ N,MIND,IH,IA,IB,IC,ID,IE,IF,IZ
      EQUIVALENCE (IB1,IB), (IC1,IC), (IZ1,IZ)
C
C     GIVEN A(IH,I) = X(I+1)-X(I), A(IA,I) = Y(I), I=1,N,
C  CALCULATE B1 = BACKWARD FIRST DIVIDED DIFFERENCE OF Y,
C             C1 = SECOND DIVIDED DIFFERENCE OF Y,
C             Z1 = FORWARD DIFFERENCE OF C1
C
C
      MAXD=MIND+2
      NM=N-1
      M=N-2
      A(IB1,1) = 0.
      DO 10 I=2,N
   10 A(IB1,I) = (A(IA,I)-A(IA,I-1))/A(IH,I-1)
      A(IC1,1) = 0.
      DO 20 I=2,NM
   20 A(IC1,I) = (A(IB1,I+1)-A(IB1,I))/(A(IH,I-1)+A(IH,I))
      A(IC1,N) = 0.
      A(IZ1,1) = 0.
      DO 30 I=2,M
   30 A(IZ1,I) = A(IC1,I+1)-A(IC1,I)
      A(IZ1,N-1)=0.
      A(IZ1,N) = 0.
      CALL SYMSOL (NRDIM,M,MIND,MAXD,IZ1,IZ,A)
C
C     GIVEN B-SPLINE COEFFICIENTS CALCULATED BY SYMSOL IN Z,
C     FIRST BACKWARD DIVIDED DIFFERENCE OF Y IN B1 AND
C     SECOND DIVIDED DIFFERENCE OF Y IN C1,
C     CALCULATE COEFFICIENTS B,C,D,E,F FOR THE QUINTIC SPLINE
C        S(I) = Y(I)+B(I)*T+C(I)*T**2+D(I)*T**3+E(I)*T**4+F(I)*T**5
C           FOR I=1,N    T=X-X(I), 0.LE.T.LT.A(IH,I)
C
      A(IZ,1)=0.
      A(IZ,NM) = 0.
      A(IZ,N)  = 0.
      Q = A(IH,1)
      Q3 = Q**3
      R = A(IH,2)
      R3 = R**3
      QR = Q+R
      H=A(IZ,2)/QR
      A(ID,1)=0.
      A(IE,1)=0.
      A(IF,1)=H/Q
      DO 40 I=2,NM
      P=Q
      P3=Q3
      Q=R
```

```
******  SPLN5B  ******

                    Q3=R3
                    PQ=QR
                    R=A(IH,I+1)
                    R3=R**3
                    QR=Q+R
                    G=H
                    H=(A(IZ,I+1) - A(IZ,I))/QR
                    A(ID,I)=10./PQ*(A(IZ,I-1)*Q + A(IZ,I)*P)
                    A(IE,I)=5.*G
                    A(IF,I)=(H-G)/Q
                    A(IB,I) = (P*A(IB1,I+1)+Q*A(IB1,I))/PQ-P*Q*(A(ID,I)-(P-Q)*A(IE,I)
                   . +(P3*A(IF,I-1)+Q3*A(IF,I))/PQ)
                    A(IC,I) = A(IC1,I)+A(ID,I)*(P-Q)-(A(IE,I)*(P3+Q3)-A(IF,I-1)*P3*P
                   . +A(IF,I)*Q3*Q)/PQ
                 40 CONTINUE
                    P=A(IH,1)
                    P3=P**3
                    A(IC,1)=A(IC,2)-10.*A(IF,1)*P3
                    A(IB,1)=A(IB,2) - (A(IC,1)+A(IF,1)*P3)*P
                    A(IC,N)=A(IC,NM)+10.*A(IF,NM)*Q3
                    A(IB,N)=A(IB,N) -  (A(IC,N)-A(IF,NM)*Q3)*Q
                    A(ID,N) = 0.
                    A(IE,N) = 0.
                    A(IF,N) = 0.
                    RETURN
                    END
```

****** SETX ******

```
      SUBROUTINE SETX
      COMMON A(30,50)
      COMMON /N1/ N,MIND,IH,IA,IB,IC,ID,IE,IF,IZ
      COMMON /N2/ IW,IX,IGP,IGM,IGBB,IGB,IG,IRBB,IRB,IR,IRP,IMU,IAB,
     . IALFHA,IYP,II
C
C     GIVEN W(J)=BEAM POWER BETWEEN R(J) AND R(J+1), J=1,N
C     CALCULATE X(J)=-LOG(1.-S(J)/S(N+1)) WHERE S(J)=SUM W(I), I=1,J-1
C     AND H(J)=X(J+1)-X(J), J=1,N-1.  SET UP AND FACTOR SPLINE MATRIX.
C     A(II,J)=DS/DX=STOT*EXP(-X)
C
      REAL A(NRDIM,N)
      STOT = 0.
      DO 10 J=1,N
   10 STOT=STOT+A(IW,J)
      A(IX,1)=0.
      A(II,1)=STOT
      S=A(IW,1)
      DO 20 J=2,N
      EXPMX=1.-S/STOT
      A(IX,J)=-ALOG(EXPMX)
      A(II,J)=STOT*EXPMX
      A(IH,J-1)=A(IX,J)-A(IX,J-1)
   20 S=S+A(IW,J)
      CALL SPLN5A(NRDIM,N,MIND,IH,A)
      RETURN
      END
```

****** STEP ******

```
      SUBROUTINE STEP
      COMMON A(30,50)
      COMMON /N1/ N,MIND,IH,IA,IB,IC,ID,IE,IF,IZ
      COMMON /N2/ IW,IX,IGP,IGM,IGBB,IGB,IG,IRBB,IRB,IR,IRP,IMU,IAB,
     . IALPHA,IYP,II
      COMMON /C/ Z,HB,HF,T,AKZ,AKZ2
C
      CALL KVAL
      CALL ALPHA
C     ADVANCE Z,GBB,GB,RBB,RB,W
      DO 10 J=1,N
      A(IRBB,J)=A(IRB,J)
      A(IRB,J)=A(IR,J)
      A(IGBB,J)=A(IGB,J)
      A(IGB,J)=A(IG,J)
      Z=Z+HF
   10 A(IW,J)=A(IW,J) * EXP(-A(IALPHA,J)*HF)
      CALL SETX
      REPS=1.E-3
      ITER=0
   15 CALL GRADP
      CALL GRADM
      DO 20 J=1,N
      A(IG,J)=A(IGP,J) + A(IGM,J)
      A(IRP,J)=A(IR,J)
      CALL DIFS (HB,HF,A(IGBB,J),A(IGB,J),A(IG,J),A(IRBB,J),A(IRB,J),
     . A(IR,J),ITER)
   20 CONTINUE
      DO 25 J=1,N
      IF(ABS(A(IR,J)-A(IRP,J)).LT.REPS) GO TO 25
      ITER=ITER+1
      IF(ITER.LT.20) GO TO 15
      WRITE(6,1000) J
 1000 FORMAT(' STEP NONCONVERGENCE, J=',I6)
      CALL EDIT
      STOP 'STEP'
   25 CONTINUE
C     INTENSITY I=DS/DY=(DS/DX)/(DY/DX)
      DO 30 J=1,N
      A(II,J)=A(II,J)/A(IYP,J)
   30 CONTINUE
      CALL EDIT
      RETURN
      END
```

```
****** SYMFAC ******

      SUBROUTINE SYMFAC (NRDIM, M,MIND,MAXD,A,R)
C
C     FACTORIZATION OF POSITIVE DEFINITE SYMMETRIC BANDED MATRIX
C       A = L*D*U WITH L UNIT LOWER TRIANGULAR, D DIAGONAL, U TRANSPOSE
C         OF L.  CF RALSTON, WILF MATHEMATICAL METHODS FOR DIGITAL
C         COMPUTERS, V2, P72.  THIS IS A MODIFIED CHOLESKY METHOD.
C     THE INPUT MATRIX IS STORED IN A BLOCK WITH NC=M COLUMNS AND
C        MAXD-MIND+1 ROWS, THE LATTER CORRESPONDING TO THE LOWER MATRIX
C     DIAGONALS, WITH THE PRINCIPAL DIAGONAL IN ROW MIND.  THE ELEMENT
C        IN ROW R, COLUMN C OF THE MATRIX IS THEREFORE IN ELEMENT ID, C
C     OF THE ARRAY, WHERE ID = MIND+R-C AND R.GE.C.
C     THE ROW INDEX R IS RETURNED AS ZERO ON NORMAL COMPLETION.  IF A
C     ZERO DIAGONAL ELEMENT IS GENERATED THE ROUTINE RETURNS AT ONCE.
C     THE OUTPUT MATRICES L AND D ARE WRITTEN OVER THE INPUT MATRIX A.
C     IT IS ASSUMED THAT MAXD-MIND+1 IS AT LEAST 2 AND NC AT LEAST 2.
C
      REAL A(NRDIM,M)
      DOUBLE PRECISION S,SL
      INTEGER R,C,CMIN,CMAX
C
      NC = M
      R=1
      IF (A(MIND,1) .EQ. 0.) RETURN
      R=2
      SL = A(MIND+1,1)/A(MIND,1)
      S = A(MIND,2) - SL*A(MIND+1,1)
      A(MIND+1,1) = SL
      A(MIND,2) = S
      IF (A(MIND,2) .EQ. 0.) RETURN
      IF (NC.LT.3) GO TO 60
      DO 50 N=3,NC
      R=N
      JMIN = MAX0(1,R-MAXD+MIND)
      CMIN = JMIN+1
      CMAX = R-1
C     THE FOLLOWING STATEMENT ALLOWS APPLICATION TO TRIDIAGONAL MATRICES
      IF (CMIN .GT. CMAX) GO TO 30
      DO 20 C=CMIN,CMAX
      JMAX = C-1
      S = A(MIND+R-C,C)
      DO 10 J=JMIN,JMAX
   10 S = S-DBLE(A(MIND+R-J,J)) * DBLE(A(MIND+C-J,J))
   20 A(MIND+R-C,C) = S
   30 S = A(MIND,R)
      DO 40 J=JMIN,CMAX
      SL = A(MIND+R-J,J)/A(MIND,J)
      S = S - SL*A(MIND+R-J,J)
   40 A(MIND+R-J,J) = SL
      A(MIND,R) = S
      IF (A(MIND,R) .EQ. 0.) RETURN
   50 CONTINUE
   60 R=0
```

```
****** SYMFAC ******

  53.                   RETURN
  54.                   END
```

****** SYMSOL ******

```
      SUBROUTINE SYMSOL (NRDIM, M,MIND,MAXD,IZ1,IZ,A)
C     ARRAY A CONTAINS ON INPUT AND OUTPUT A DIAGONAL MATRIX D IN ROW MIND,
C     SUBDIAGONALS OF A UNIT LOWER TRIANGULAR MATRIX L IN ROWS MIND+1 TO
C     MAXD, VECTOR Z1 IN ROW IZ1, AND ON OUTPUT A SOLUTION VECTOR Z IN
C     ROW IZ.    THE SYSTEM SOLVED IS  L*D*U*Z=Z1 WHERE U IS THE
C        TRANSPOSE OF L.
C
      REAL A(NRDIM,M)
      DOUBLE PRECISION S
      INTEGER R
      NC=M
C     SOLVE L * DUZ = Z1  FOR  DUZ
      A(IZ,1)=A(IZ1,1)
      DO 20 R=2,NC
      KMAX=R-MAX0(1,R-MAXD+MIND)
      S = A(IZ1,R)
      DO 10 K=1,KMAX
   10 S=S-DBLE(A(MIND+K,R-K))*DBLE(A(IZ,R-K))
   20 A(IZ,R)=S
C     SOLVE  D * UZ = DUZ  FOR UZ
      DO 30 R=1,NC
   30 A(IZ,R)=A(IZ,R)/A(MIND,R)
C     SOLVE  U * Z = UZ  FOR Z
      DO 50 I=2,NC
      R=NC+1-I
      KMAX=MIN0(NC-R,MAXD-MIND)
      S=A(IZ,R)
      DO 40 K=1,KMAX
   40 S=S-DBLE(A(MIND+K,R))*DBLE(A(IZ,R+K))
   50 A(IZ,R)=S
      RETURN
      END
```

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFOSR-TR-78-0775	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) ANALYSIS OF LASER-SUPPORTED DETONATION WAVES AND LASER BEAM PROPAGATION IN TWO DIMENSIONS		5. TYPE OF REPORT & PERIOD COVERED Final 10/1/76-9/30/77
		6. PERFORMING ORG. REPORT NUMBER SSS-R-78-3564
7. AUTHOR(s) J. R. Triplett M. H. Rice		8. CONTRACT OR GRANT NUMBER(s) F49620-77-C-0006
9. PERFORMING ORGANIZATION NAME AND ADDRESS Systems, Science and Software P. O. Box 1620 La Jolla, California 92038		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 2301/A5 61102F
11. CONTROLLING OFFICE NAME AND ADDRESS Air Force Office of Scientific Research/NP Building 410, Bolling Air Force Base, Washington, D.C. 20332		12. REPORT DATE February 1978
		13. NUMBER OF PAGES 78
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Shock Wave Propagation
Laser Beam Propagation
LSD Waves
Paraxial Wave Equation

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

It is shown that the Flux-Corrected Transport (FCT) algorithm correctly computes the propagation velocity and Chapman-Jouguet pressure for a laser-supported detonation (LSD) wave, even for zone size large compared to the radiation absorption length. A two-dimensional FCT computer program for analysis of LSD wave propagation phenomena is presented. A trajectory method for analysis of two-dimensional laser beam propagation in the paraxial Fresnel

next page

approximation, with continuously varying refractive index, is described.